INVESTIGATION ON MOLECULAR DYNAMICS SIMULATION OF NANOMETRIC CUTTING

By

ROBERT LELAND STEWART

Bachelor of Science

Oklahoma State University

Stillwater, Oklahoma

1995

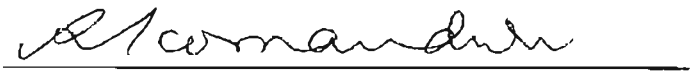
Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 1998

INVESTIGATION ON MOLECULAR

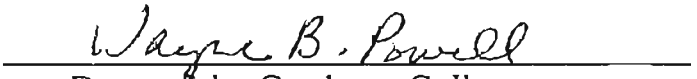
DYNAMICS SIMULATION OF

NANOMETRIC CUTTING

Thesis Approved:



Thesis Advisor



Dean of the Graduate College

ACKNOWLEDGEMENTS

I would first like to thank my advisor, Dr. Ranga Komanduri, for not only his guidance and patience during this project, but also for inspiration in the field of research. I would also like to thank Dr. Lionel Raff for many discussions and insight on molecular dynamics, as well as Dr. A. Ghajar and Dr. H. Lu for serving on my committee and providing suggestions thereof.

I would also like to thank my colleagues at the Mechanical and Aerospace Engineering Research Lab (MAERL) for their assistance on this project. The constant interaction between many individuals of varied backgrounds and interests is what makes the research activity at the MAERL so interesting. Special thanks are due to Naga Chandrasekaran and David Stokes with whom I interacted heavily on molecular dynamics related projects.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Ultra Precision Machining

Ultra Precision Machining (or UPM) is a term used for machining at the finest level possible with current methods. Today, UPM involves machining at the nanometric level. To achieve such high precision, the tool must be extremely resistant to wear and have a high degree of sharpness and absence of any grain boundaries. Thus single crystal diamond tools are used for machining of non-ferrous materials. Examples of parts made of non-ferrous materials include aluminum and copper mirrors for lasers and aluminum disc drives. The cutting tool is part of a large dedicated UPM machine, which focuses on high rigidity, high precision, minimum level of vibration, contamination, and other possible influences which could have an effect at such a level of precision. Thus the entire process is of necessity controlled by some type of feedback control system. The machine tool is generally located in a temperature and humidity controlled room. To maintain close control on the temperature, the entire machine tool is continuously showered and if possible remotely operated. This is an advantage in that the operator does not really enter into the equation when it comes to machining error but it is a disadvantage in that the proper settings for the control system need to be known before machining.

In spite of several careful studies, the mechanism of machining at the nanometric level is not fully understood. Thus, for a unique machining situation, the only course of action that is viable at this point is to combine results from similar past machining operations with several experiments on the configuration desired to be machined. However, the acquisition of several single crystal diamond tools with different rake angles along with reconfiguring the control system/machining apparatus can be extremely time consuming and expensive. Plus, the absolute best solution may not be found but only a "local maximum" solution. If the intrinsic behavior of the process was obtained, then a more thorough and fruitful analysis of a particular situation could be conducted. This would require knowledge of atomic positions and behavior, which at this time is nearly impossible to measure during a UPM operation, especially below the surface. New methods (such as computer simulation, see below) may prove to be a promising alternative to traditional empirical optimization techniques.

1.2 Applications of UPM

In today's world, technology is increasing at a very rapid pace. This is especially so in the field of electronics, and with such a pace comes a commensurate decrease in the size (and increase in required manufacturing precision) of components. There are several reasons why electronic devices are becoming smaller:

- The speed of a circuit is influenced by the total path length that the signal must travel. Reducing the path length (and thus the size of the circuit) increases the performance.
- More complex logic requires more circuitry. CPU's have millions of transistors, gates, and other components all competing for a small space. The spacing of that circuitry must be precise to reduce chances of "cross-talk" and improve heat transfer.
- Optical devices require a high degree of accuracy to minimize undesired diffraction of signal due to surface imperfections

Another field which continually demands improved machining precision is that of micro/nano machines. These devices have a wide range of possible applications including surgical/medical tools, space applications (due to their extremely low payload weight), and ultra precise motion controllers.

As the scale of UPM approaches that of machining materials atom-by-atom, the need for characterization of interatomic behavior under machining conditions becomes more and more important. This goes way beyond the continuum mechanics of macro-scale machining, and requires an accurate model of the bonds/forces present at this level. This requires massive computational power, towards which the recent strides in computer

technology could prove to be the solution.

1.3 Molecular Dynamics Simulation

Molecular dynamics (MD) simulation is the principle of using various equations that describe interatomic forces to analyze the behavior of sets of atoms under varying conditions. The simulations themselves are conducted using a computer that iterates through the many computations necessary to resolve the trajectories. The resulting data can then be displayed in numerical (forces, potential energies) and visual (animation, still pictures) form. The prospect for characterizing intrinsic atomic behavior by virtue of being able to see it happen is awesome. Perhaps the single most important advantage of having an accurate computer simulation technique is not to test known configurations but to be able to try configurations previously not thought of. Many of the recent developments in materials research have been greatly affected by the ability of chemists and material scientists to investigate the possibility of creating a new material *before* actually creating it. The case in point is the possibility of creating carbon boron nitride, which is expected to be harder than diamond, if only it can be successfully synthesized. This applies not only to new material creation but also to related fields such as research conducted into diseases. Molecular biologists are able to try out a particular structure of a new antigen or the like to see if it is viable.

1.4 MD and its Relationship to UPM

In order to perform ultraprecision machining, a high degree of control is necessary over the parameters during cutting. As mentioned above, small deviations due to vibration or other influences can severely affect the results. Thus, elaborate control systems are implemented to control the forces, vibrations, etc. on the tool during cutting. If the desired shape of the tool and cutting parameters are known, they can be controlled somewhat easily. However, the mechanism of cutting at the nanometric level is still largely unclear, and thus standard machining principles may not be applicable in these situations. As mentioned above, one possible way to determine the optimal parameters for a particular UPM application would be to try different tool shapes, forces, control algorithms, etc., but this would prove very expensive in terms of time and equipment. If the process could be simulated accurately by a computer, then the optimal configuration could be determined "virtually", without a single physical experiment performed.

In order to perform nanometric cutting, the mechanism of interaction between atoms must be known. The field of Molecular Dynamics (MD) is concerned with just such a behavior. As mentioned above, it attempts to quantify the interactions between atoms into sets of equations that can be solved using various approaches. The computational speed of computers today is nearing the point at which large enough sets of atoms (on the order of a few millions) can be simulated so as to identify UPM behavior over a few hundred nanometers of material. The key issue is to determine what

the current deficiencies are in the mathematical models of atomic interaction so these simulations can be deemed valid. Once an accurate model is achieved, the practice of "virtual machining" mentioned above will almost definitely be utilized in the UPM field.

CHAPTER 2

LITERATURE REVIEW

2.1 Ultra Precision Machining

The field of Ultra Precision Machining (UPM) lies at the extreme limit of machining accuracy. Since technology is continuously moving forward, the tolerance range of UPM is constantly being reduced. Machining processes have of late been characterized into three "levels": normal or conventional machining (CM), precision machining (PM), and ultraprecision machining (UPM). The progression in these three areas throughout this century is shown in Figure 1. It can be seen that with time, yesterday's UPM is becoming today's PM, and yesterdays PM's is becoming today's CM.

2.2 Machine Tools for UPM

The machine tools for ultra-precision machining using single crystal diamond were developed in the 1970's at Lawrence Livermore National Laboratory (LLNL) with primary focus placed upon optical applications for the defense industry. A diamond turning machine for large diameter optics was developed subsequently also at LLNL by

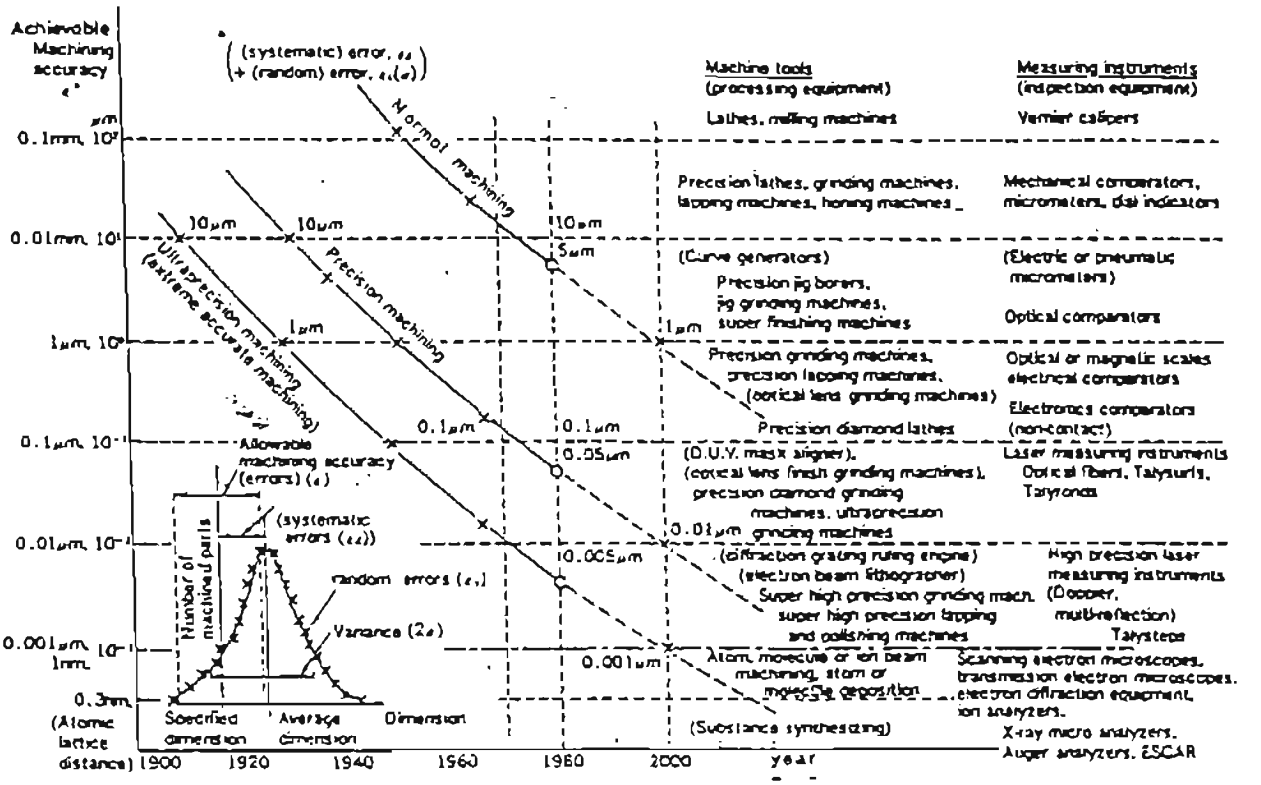


FIGURE 1 : History of machining accuracy (after Taniguchi, 1983)

Donaldson (1979). Precise temperature control is a pre-requisite in UPM to avoid deviations due to thermal expansion, etc., and thus Donaldson's machine paid particular attention to this by operating the machine remotely in a temperature controlled room and by implementing temperature control on the machine tool components. A 3-axis general purpose ultra precision grinding (UPG) machine was designed and built by at CUPE (Cranfield Unit of Precision Engineering) under the leadership of Professor McKeown (1990). It combined diamond turning, grinding, polishing, and measuring into one device capable of 1.25 nm accuracy. A UPG machine to produce aspherical optics was produced by Ueda (1991) which used a fine grained resinoid bonded diamond grinding wheel. Surface roughnesses of 0.6 nm R_a were achieved with this machine. Finishing of large wafers for the electronics industry was accomplished by an ultraprecision float polishing machine developed by Namba (1987). Subsequent work by Namba and Abe (1993) produced a technique of grinding glasses with resinoid bonded diamond grinding wheels to a surface roughness of about 0.15 nm. This displayed the capability of producing ultrasmooth surfaces with grinding alone (no polishing). Ando (1992) produced a super smooth polisher which could polish 500 mm diameter optics to a contour accuracy of 78 nm and surface roughness of 0.13 nm rms.

In all of the above machine tools, attention was paid to factors such as temperature control, vibration damping, and control system accuracy. As mentioned with the Donaldson machine, controlling the temperature can be critical to produce good results, and thus various methods such as oil baths and even the development of a zero-thermal expansion glass-ceramic spindle (Namba, 1989) were employed to minimize

temperature effects. Vibration was often combated by using very heavy and stiff materials (granite-epoxy or the like) as well as methods to decouple any moving parts from the critical interface sections. Accurate control systems were vital to the above machines as well, and so laser interferometry techniques were employed to give maximum performance.

2.3 Mechanisms of UPM

When machining at small depths of cut, there is a noticeable deviation from standard machining behavior. This is most likely manifested by the fact that the relative shape of the tool tip, shear zone size, etc. are approaching the size of the atomic structure of the workpiece/tool, and thus more relatively important towards one another.

Nakayama and Tamura (1968) observed a definite increase in specific energy with tool edge radii and decrease in depth of cut. The scale of this research was limited to a few micrometers cut depth as ultraprecision machine tools were not available. Furukawa and Morunuki (1988) also found an increase in specific energy below a certain depth of cut (around 3 μ m). Again this was attributed to the fact that the tool tip radius and other dimensions were becoming large as compared to the depth of cut. In effect, the tool can be considered to be blunt with respect to the workpiece when the depth of cut is less than the edge radius of the tool. This accounts for the difference in the behavior of

the same tool with the same workpiece yet smaller depth of cut. Figure 2 shows the relative progression from a standard positive rake machining operation to, grinding, to ultraprecision machining, to an indentation-sliding process after Komanduri et al (1997). This is akin to what happens as the depth of cut is decreased to below the edge radius of a tool. Of course, if an atomically sharp tool were available, the comparison between the behavior at ultraprecision level and larger scale machining would likely be more similar.

Rubbing at the flank face of the tool due to elastic recovery of the machined surface also can contribute to deviations in the behavior at small depths of cut. Morikawa and Okuda (1989) observed the specific energy to increase with decreasing depth of cut and attributed this behavior to this rubbing at the flank face. Thus, as depth of cut is decreased, the effect can be described more in terms of plowing than conventional machining, as the effective rake angle is increasing and there is a large amount of contact with the tool surface which previously was not influencing the behavior. This increase in specific energy coupled with a change in deformation characteristics imposes a lower limit on the depth of cut for a particular tool shape. The more accurate a tool is to the atomic scale, the more it can prolong this transition to different behavior. Tool edge radius can be produced accurately to around 20nm today but that is still much larger than what is considered to be a super-smooth surface (1 nm or less).

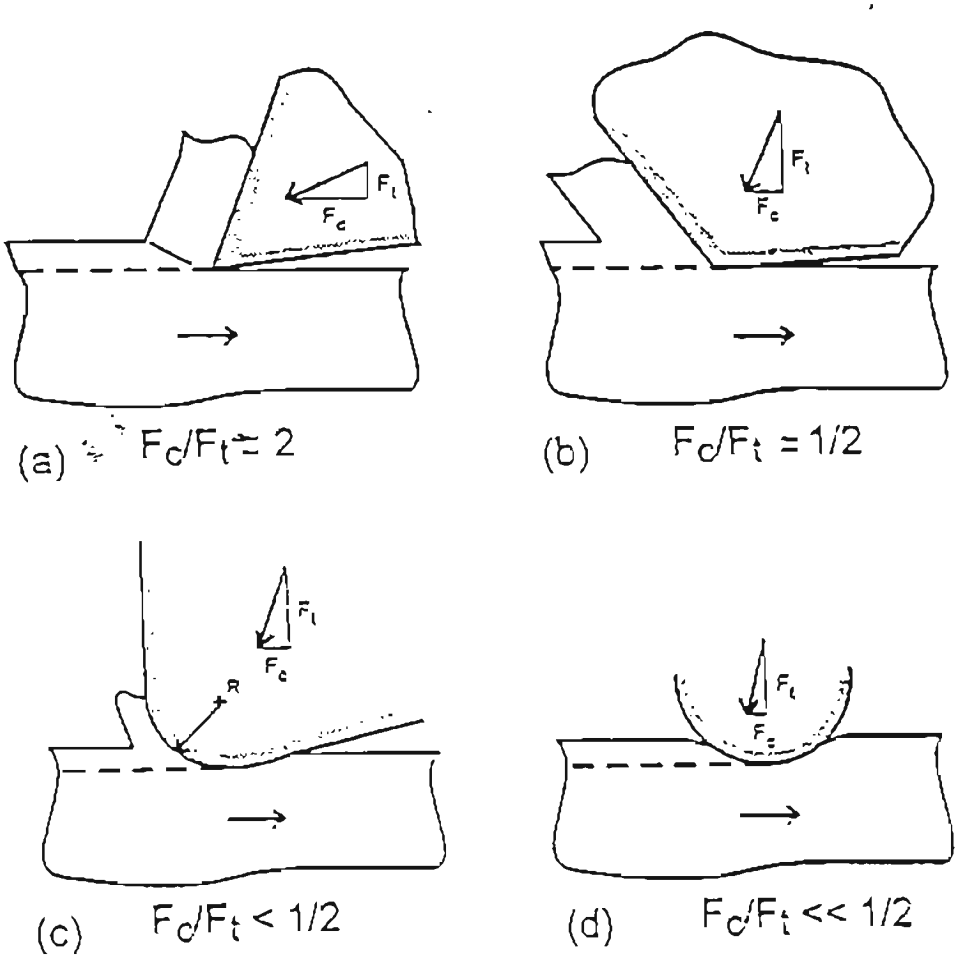


FIGURE 2 : Tool/Workpiece interactions with change in tool geometry: (a) conventional cutting (b) grinding, (c) ultraprecision machining at small depths of cut, and (d) indentation sliding (after Komanduri and Lucca, 1997)

Lucca et al (1991) studied this trend with decreasing depths of cut with regards to the ratio of cutting force to thrust force. As expected, overall forces decreased with decrease in depth of cut down to about 2 μ m. Specific energy then began to increase with decreasing depth of cut, as well as the ratio of thrust force to cutting force. The increases in forces were attributed to increased tool/work interaction due to the small depth of cut.

In all of the above cases there was a definite trend towards increase in specific energy with decrease in depth of cut below a certain level. While force and energy data are important and can reveal much about a particular process, visualization of the cutting process as it happens could provide crucial information about the nature of the deformation process and interaction between the tool and the work. Today there are several highly accurate means of direct analysis of work materials, as shown in Figure 3.

It is possible with all of these techniques to observe a workpiece's structure before and after a machining process, and to a limited degree during machining itself. One major limitation is that actual atomic structure below the surface cannot be observed during the process, and being able to see "into" the workpiece during machining could provide invaluable information as the underlying process of UPM. The possible solution to this dilemma is the use of computer simulations, where all data (including atoms "inside the workpiece") is readily available for analysis. Thus implementing MD simulations of UPM is an attractive research possibility.

Instrument	Resolution (nm)		Range (mm)	
	Lateral	Vertical	Lateral	Vertical
Stylus	100-250	0.3	>100	1
Interferometric microscope	500	0.1	7	0.1
AFM	2	<0.1	0.1	0.005
STM	2.5	0.2	0.1	0.0001
Nomarski microscope	>500	—	—	—
SEM	10	2	—	0.002
TEM	2	2,000	—	0.0001

FIGURE 3 : Resolutions and range of some surface scanning instruments (after Lonardo et al., 1996; Whitehouse, 1994)

2.4 Molecular Dynamics Simulation

The advent of computers has opened up a world of computational research previously unavailable to mankind due to incredibly large amounts of mathematical operations. Molecular Dynamics (MD) is one such field. The pioneering work in this area was conducted at the Lawrence Radiation Laboratory (LRL) by Alder and Wainwright (1959). The growth of MD simulation has been a joint effort between chemists and physicists continuously developing new potentials to attempt to describe complex interatomic behavior, and the increasing computational power of computers which allows for those potentials to be tested with larger and larger atom sets. Many applications of MD exist and have been the subject of continued progress (Hoover 1986, Levine 1987, Allen and Tildesley 1991). Due to the fact that potentials are developed primarily by studying small atom sets, their behavior when "scaled up" to larger atom sets is where discrepancies usually manifest themselves. This is especially true when applying MD simulation to machining processes. The data acquired from MD simulations of UPM as compared to known empirical data will not only help researchers to better understand the mechanism of UPM, but also to help refine the atomic potentials developed for use in MD simulation.

2.5 MD Simulations of UPM

Only recently have computers acquired the necessary horsepower, in terms of speed and memory capacity, to conduct MD simulations of any significant size with respect to machining. This power is necessary due to the fact that the number of atoms involved in a typical UPM process can well reach into the millions and beyond, and at least a few thousand are necessary for any kind of machining behavior to be observed. So, not only the computational speed requirements but also memory capacity requirements can become quite large. Fortunately, progress in both areas in the realm of computer hardware is quite rapid. Other advancements such as parallel processing and UMA (unified memory architecture) are allowing a greater amount of atoms to be simulated in less time. Thus work is beginning to increase in fields such as MD simulation of UPM.

Belak et al (1990) pioneered MD simulations of nanometric cutting of copper at LLNL. These were conducted in both 2D (to save computation time and focus on planar behavior) and quasi -3D (to allow for real behavior) fashion. The interatomic potential used was the embedded atom method, which uses functions of background electron density to relate forces on atoms. Tests were performed on copper workpieces using an infinitely hard tool, i. e. the tool atoms were not allowed to deform during the simulation. Cutting speed used was significantly higher than in practice (on the order of 100 m/sec) in order to save computational time, which can affect the results. These simulations behaved similar to the experimental machinining results in that the specific energy

increased with decreasing depths of cut. Larger radii tools also experienced larger forces for the same depth of cut as smaller radii tools. Shimada et al (1992) observed similar results in 2D MD simulations of nanometric cutting of copper. Studies were also conducted by Shimada et al (1993) on minimum thickness of cut (MTC) below which machining accuracy is lost. Both copper and aluminum were simulated with various radii tools, and the trends indicated that copper possessed a lower MTC for the same tool geometry than aluminum. The interaction potentials between tool and work could be the significant reason for this, as well as the respective plasticity of each material. Surfaces on the aluminum were observed to be rougher as well after machining, also attributable to the aluminum's higher plastic deformation and diamond affinity.

It is clear that MD simulation hinges upon the accuracy of the interatomic potentials used. Potentials are generally well developed for like-to-like materials (atom bonding with another atom of the same element). However, unlike bonding is usually largely undefined, especially for the simpler potentials. Inamura (1992) investigated the effect of the "unlike" potential upon several MD simulations of nanometric cutting of copper by an infinitely hard tool. When a Morse potential (interatomic potential between the tool and work possessed both attractive and repulsive components, i.e. chemically active) was used for the interface potential, there was a large amount of clustered atoms observed to gather in front of the tool during machining. This is due to the attractive component of the interatomic potential between the tool and work. When a Born-Meyer (repulsive behavior only, i.e. inert) potential was used, machining behavior was more in line with that observed during conventional machining (shear zones, standard chip

formation, etc.) Thus it would seem that unless chemical interaction between the tool and the work is known to be significant then an inert (Born-Meyer) approach is appropriate. This effectively makes MD simulation the study of disturbing the workpiece by a moving geometry of repulsive atoms (the tool). Certainly there is some interaction between tool atoms and work atoms during real machining, especially at the high temperatures generated during such processes. As work progresses in the development of appropriate potentials, proper adjustment to MD simulations should be made accordingly.

Inamura et al (1993) also investigated the force and energy variations during MD simulation of nanometric cutting. The forces on the tool dropped significantly at several points during machining for short times. This was attributed to the dislocations generated in the workmaterial by the tool, as once the dislocations began moving the force exerted by the section of workpiece inline with it upon the tool decreased. Potential energies of each atom during the machining process were examined as well. About half the energy dissipation was observed in the shear zones, a little less due to plastic deformation below the tool, and the remaining small amount by new surface generation. The amount of shear stress in the shear zone was also observed to be less than the theoretical yield stress, and so implied a secondary mechanism of deformation.

Maekawa et al (1995) continued the analysis of interface potential effects by simulating nanometric cutting of copper using both chemically active (attractive and repulsive) and inert (repulsive) potentials. When active potentials were used, there was a

large tendency of the workpiece atoms to migrate towards the tool, which was likened more to energy beam processing than conventional machining. Inert potentials did not produce chips and resulted in more of an extrusion behavior. The tool was altered from being near infinitely hard to less cohesive, in order to study possible tool wear mechanisms. It appeared that the mechanism was one of continuous tool-work atom diffusion along with worn tool particles re-adhering to the tool.

At OSU considerable work is being conducted on MD simulation of nanometric cutting and tribology. Chandrasekaran et al (1998) developed a new method of MD simulation, called length restricted molecular dynamic simulation to enable large lengths of workpieces to be machined without the need for a large number of atoms. Thus processing times could be reduced considerably and the memory requirements could also be reduced. Komanduri et al (1997) investigated the effect of high negative rake angles to simulate grinding. Komanduri et al (1998) also investigated the effect of edge radius on nanometric cutting. Komanduri et al (1998) also investigated the effect crystal orientation and direction of cutting in nanometric cutting of single crystal workmaterials. In the area of tribology, Komanduri et al (1998) investigated nanoindentation and scratching as well as nanometric friction in sliding (1998). Several other studies both on nanometric cutting and nanotribology are in progress. They include the effect of different workmaterials, entry and exit effects, workmaterial-tool combinations of different hardness levels, and multiple tool cutting, such as milling.

From the review of literature and work conducted upon MD simulation of nanometric cutting upto this point, several issues became apparent to the prospective MD researcher:

1) The need for more computational speed, whether garnered by faster computers or through other means (algorithm refinement, etc.), is an important aspect.

2) Using different potentials for the interaction of unlike materials can highly affect the results of an MD simulation.

3) Due to the uncertainty of interatomic potential accuracy, the goals at this stage of MD simulation are as much to verify an interatomic potential's validity in large scale situations as it is to relate the observed behavior to the empirical realm.

4) The need to develop MD simulation software that is more user friendly.

5) The need to improve visualization such as still pictures of MD simulation as well as animation.

CHAPTER 3

PROBLEM STATEMENT

Molecular dynamics modeling of nanometric cutting requires the integration of several different fields of expertise. Understanding of how to utilize interatomic potentials requires knowledge of chemistry and physics. Solving the equations of motion for the atoms requires knowledge of mathematics. Coding a molecular dynamics program so as to be sufficiently useful and fast requires application of computer science. Finally, simulations need to be created and evaluated using knowledge of engineering.

Therefore, a broad understanding of all the aspects of MD simulation is required. This will allow for various modifications to be made to the process to improve the performance as well as aid in the analysis of results. As the goal of MD simulation is to mirror reality, fundamental knowledge of both real world behavior (experimental results) and MD's underlying behavior are vital in order to relate the two.

There are two main limiting factors to MD simulation today. The first is the lack of an all encompassing interatomic potential behavior. While results have been somewhat promising to this point (as discussed in Chapter 2), much work has yet to be performed to better define the interactions of materials so they behave more realistically. Simulating

various processes where the outcomes are generally known will aid in this process, as macro behavior often is hard to extrapolate from micro behavior. The other limiting factor to MD is computational speed. When simulating three dimensional sets of atoms, the amount of atoms and more importantly the amount of bonds present can increase at an alarming rate. For example, a 40 nm x 40 nm x 40 nm cube of FCC atoms with a lattice constant of 0.4 nm contains 4,000,000 atoms. If each of these atoms is interacting in a non-negligible fashion with 10 neighboring atoms, the amount of bonds present is approximately 20,000,000. This obviously can lead to severe computational loads, which slow the MD process considerably. Any method that would reduce the computational time without sacrificing significant accuracy should be looked into and exploited.

The goals of this investigation include the following:

1) To investigate various optimizations approaches, namely, in the realms of runtime and error correction. Test various approaches and investigagte how much improvement can be obtained. MD simulations are highly intensive computationally and can result in extremely long runtimes for large atom sets. The number of computations is also typically very large and so precision is important to maintain.

2) Develop a method for setting up molecular dynamics simulations, so as to provide sufficient flexibility in terms of initial simulation geometry. Also, investigate and develop

a method to view molecular dynamics data. In order to conduct MD simulations, the initial positions and other characteristics (such as material type, etc.) need to be defined for the atom set in question. For simple geometries (such as box shapes) this is not very difficult but more complex geometries (such as polycrystalline materials) present somewhat of a problem.

3) Using the approaches developed above, evaluate the effectiveness of the methods used for grain initialization, grain size, grain orientation, and grain boundary shape have upon MD simulations and compare it to known material behavior.

As discussed earlier, cutting at the nanometric level differs from larger scale cutting due to the fact that the interaction is now more of a particle nature than a continuum. The conventional machining wisdom which applies at larger scales may not always hold good for nanometric cutting. Divergence from conventional behavior is also found in nanocrystalline materials versus those with larger grain sizes.

1. To develop the required software to enable animations of the MD simulations of various nanometric cutting and tribological processes.
2. To develop a framework for MD-CAD for MD simulation of nanometric and tribological problems. This will facilitate user friendly approach to MD simulation of nanometric cutting.

CHAPTER 4

PRINCIPLES OF MD SIMULATION

The basic principle of molecular dynamics (MD) is to repeatedly solve the Newtonian equations of motion for the desired system of atoms. This requires two things: a simultaneous solution method of sufficient accuracy as to allow this approach to be valid and a method of generating the forces between the atoms.

4.1 Solving Newtonian Equations for MD

Suppose that there is a system of N particles with characteristics in three dimensional space denoted by :

Mass of ith atom = $m_i(t)$ (1)

Position of the ith atom at time t = $r_i(t)$ = x,y,z vector (2)

Velocity of the ith atom at time t = $\dot{r}_i(t) = v_i(t)$ (3)

Acceleration of the ith atom at time t = $\ddot{r}_i(t) = \dot{v}_i(t) = a_i(t)$ (4)

These equations lead to :

$$\text{Momentum of the ith atom at time } t = m_i(t)v_i(t) = p_i(t) \qquad (5)$$

$$\text{Force on the ith atom at time } t = m_i(t)a_i(t) = F_i(t) \qquad (6)$$

Using Newton's second law, the force on each atom is equal to its rate of change of momentum, namely:

$$F_i(t) = \dot{p}_i(t) \qquad (7)$$

Assuming that a system of particles has the current forces and momentums for each atom defined, one can digitally step the system to a new position by applying the above equations over a discrete time step. This "Euler method" approach though simple, can lead to inaccuracies. A more accurate method, which is commonly used to solve such simultaneous systems, is the fourth order Runge-Kutta method. It attempts to evaluate the average forces experienced over a time step by calculating forces at 4 different points. The method is outlined below for a single particle:

For N particles with initial derivative D1 and desired time step t:

1) move to a new position P2 using D1 and a time step of t/2

2) calculate new derivative D2 using forces experienced at P2

3) return to position P1

4) apply D2 over a time step of t/2 to arrive at a new position P3

5) calculate new derivative D3 using forces experienced at P3

6) return to position P1

7) apply D3 over a time step of t to arrive at a new position P4

8) calculate new derivative D4 at this point

9) return to position P1

10) calculate $D_{avg}=(D1+2*D2+2*D3+D4)$

11) use this D_{avg} and a time step of t to advance the atom to its final position

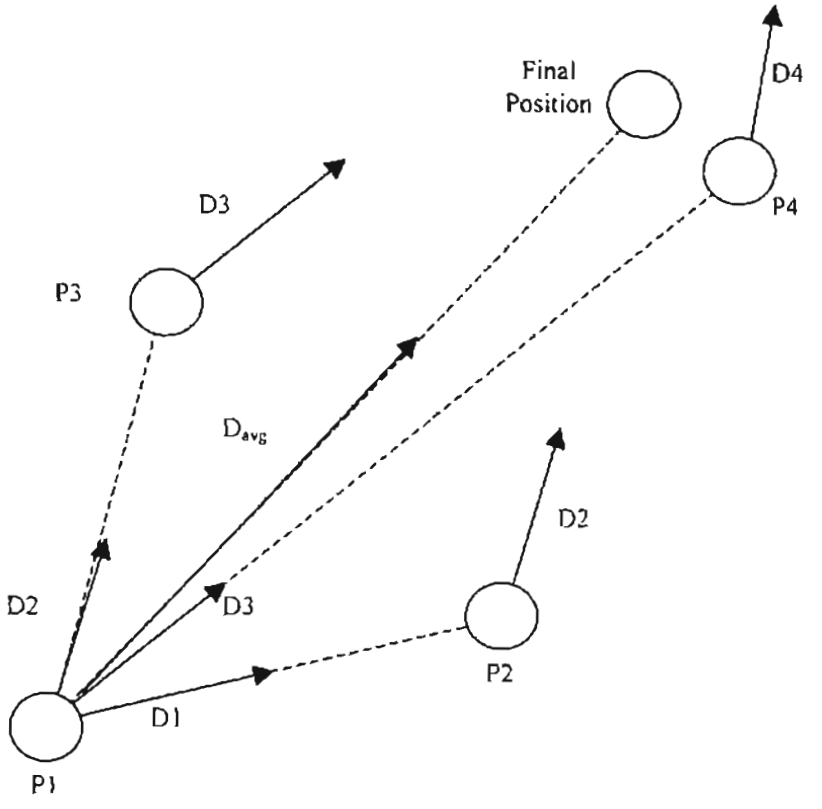


Fig 4 : Diagram of Runge-Kutta method in action

Figure 4 elucidates this procedure. It is important to note that this is a simultaneous solution approach, in that all particles are moved together during steps 1-11 above. Thus for a set of N particles, there will be 4N intermediately calculated positions and forces for a Runge-Kutta procedure per time step.

So basically the Runge-Kutta method weights the final behavior more heavily upon "midway" calculations than those at the ends. It is fairly accurate, but as can be seen from above, it effectively solves the system four times before incrementing the atom position permanently. This can be time consuming, but the accuracy gained is worth the computational cost. The Runge-Kutta method is also a "self-starting" technique, in that it can be applied to a set of atoms that "instantly appear" (as is often the case in molecular dynamics simulations, i.e. the atoms all start together at some initial time). Some other simultaneous solution techniques (such as predictor-corrector methods) are not self-starting, and require a startup routine (such as Runge-Kutta) to gather sufficient data history. The amount of derivatives evaluated per time step is generally less for the predictor-corrector methods, however a smaller time step is often required in predictor-corrector method to achieve similar accuracy, thus evening out computational time. Therefore, Runge-Kutta is a good choice for such calculations.

4.2 Calculating Interatomic Forces

In MD simulations, a method for calculating the forces generated by each atom upon the others is necessary in order to have Newtonian equations to solve in the first place. Typically, the interatomic behavior is defined in terms of potential functions. These functions effectively define the potential energy present between two atoms, from which can be derived forces (the derivatives of potential). They can be very complex to very simple in scope. The simpler functions are based solely upon characteristics of the two atoms in question (their respective positions, material types, etc.). The more complex potentials introduce terms based upon other atoms' positions, bond angles with other atoms, etc., which help to correct for behavior that can be related to those factors. Silicon, for instance, has a complex behavior that requires more than a "two-body" potential force can simulate (Stillinger and Weber 1985). However, simple BCC and FCC metal structures have been simulated fairly successfully with pairwise potentials, and these potentials (specifically Morse) are used in the later described simulations.

4.3 Pairwise/Morse Potentials

These functions attempt to define a potential energy relative to the position of two atoms. The Morse potential, a simple pairwise potential, is defined as follows:

$$V(r) = D e^{-2\alpha(r-r_e)} - 2D e^{-\alpha(r-r_e)} \qquad (8)$$

where V(r) potential at radius r

r = radial distance of the two atoms

r_e = "equilibrium radius"

Note that the equation:

$$Fx_i(t) = -\frac{\partial V_i(t)}{\partial x} \qquad (9)$$

where $V_i(t)$ is the potential of particle i at time t and Fx is the force in the x direction and its y and z counterparts is used to relate potential energy to forces.

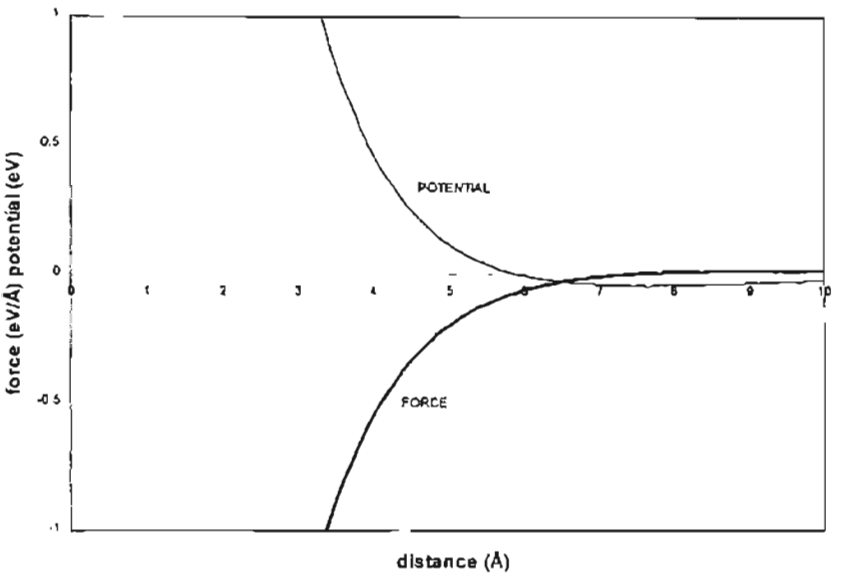


Figure 5a : Morse potential curves for cesium

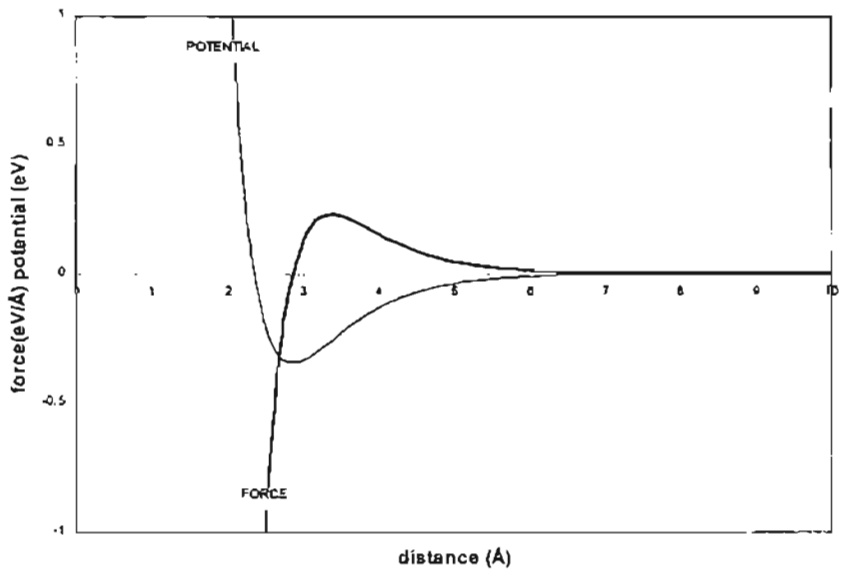


Figure 5b : Morse potential curves for copper

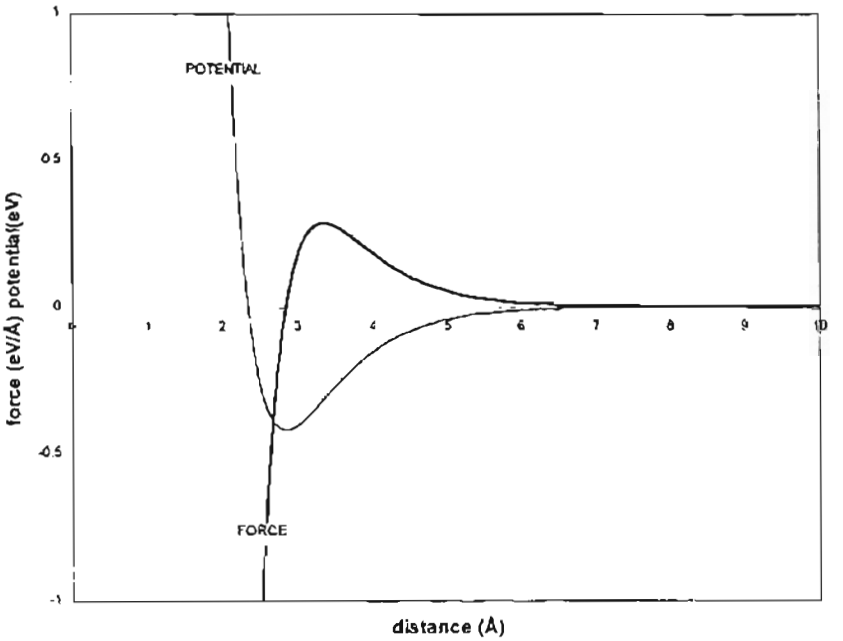


Figure 5c : Morse potential curves for iron

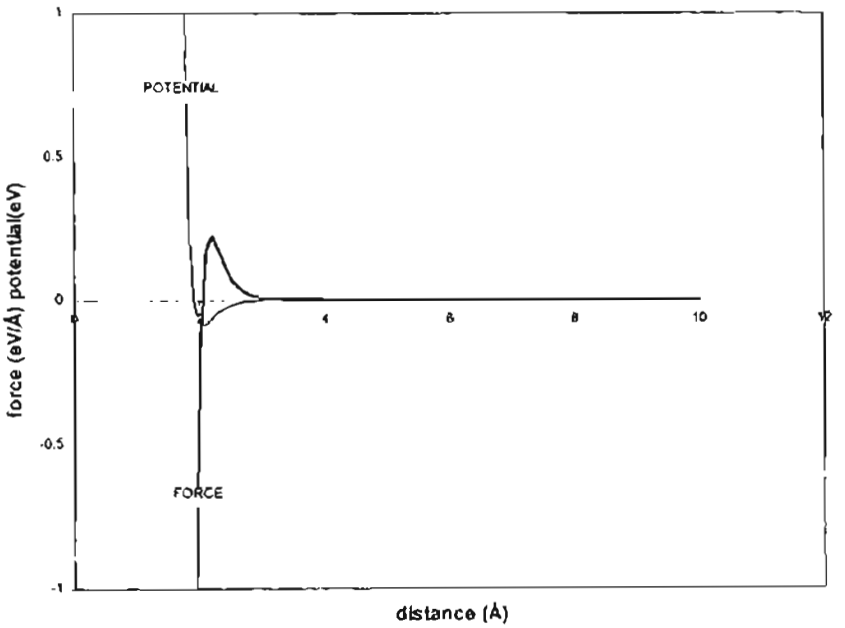


Figure 5d : typical "neutral" Morse potential, i.e. low chemical affinity

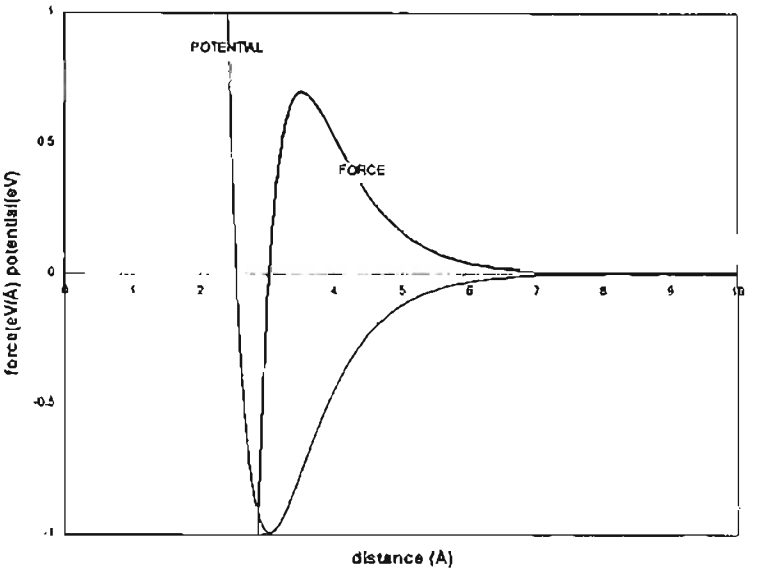


Figure 5e : Morse potential curves for tungsten

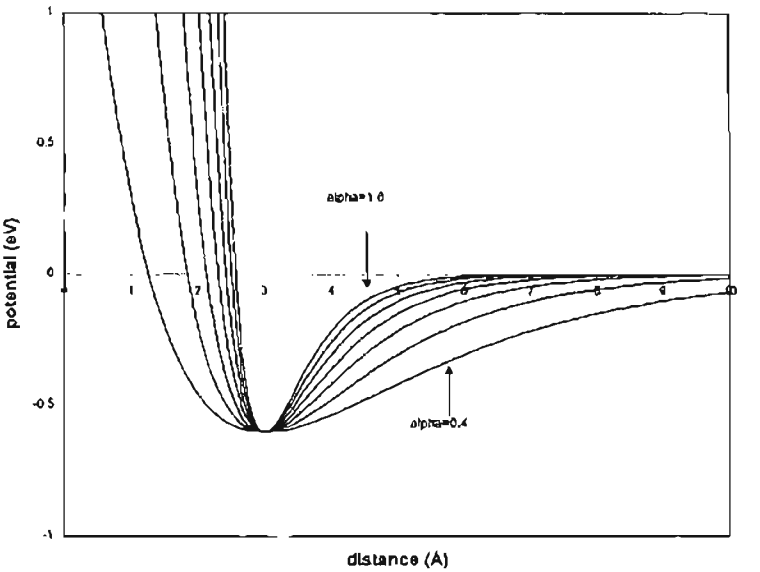


Figure 5f : variance of Morse potentials with alpha

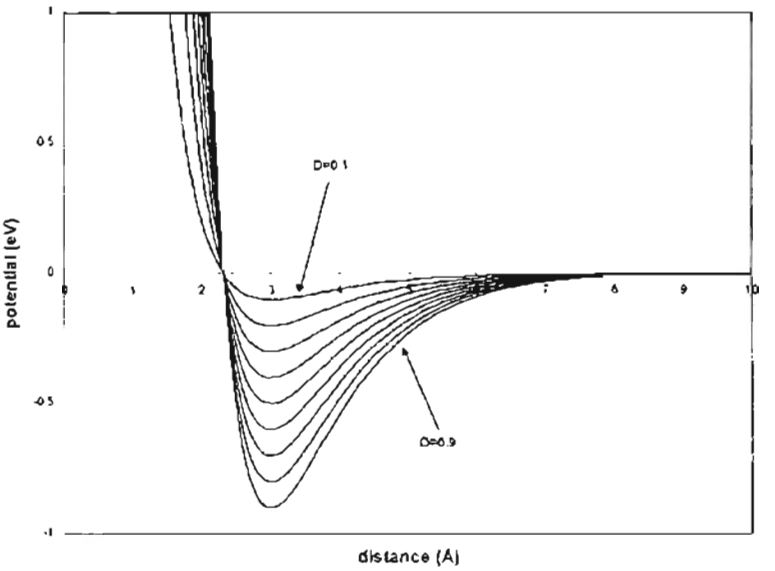


Figure 5g : Variation of Morse potentials with D

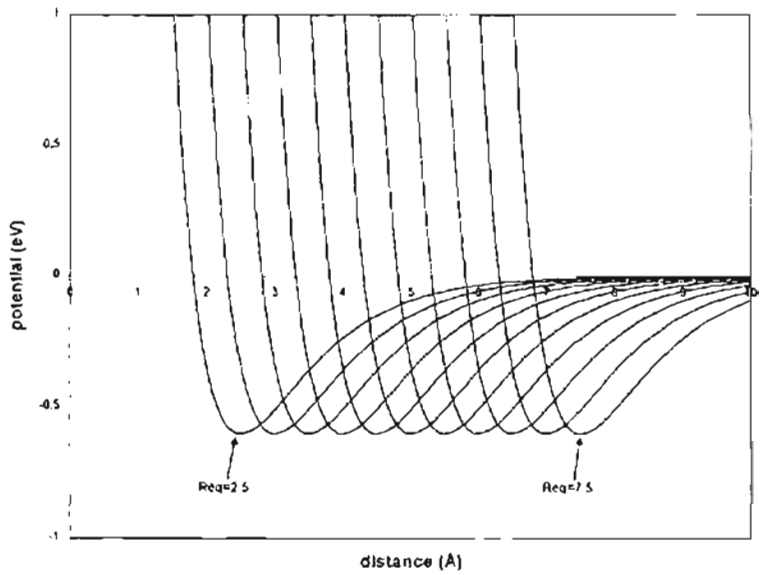


Figure 5h : variation of Morse potentials with Req

This potential equation has two terms - an attractive term and a repulsive term. The first term is repulsive, in that potential energy increases with decreasing radius below r_e. This simulates the fact that two atoms will repel below their "equilibrium" separation radius (r_e). The second term is attractive in that potential energy decreases with decreasing r. This term dominates when r is larger than r_e, and simulates the attractive force two atoms possess when their separation is larger than r_e. Both terms reduce to zero for large r, simulating the negligible interactions of two atoms separated by a large amount. The parameters D, α, and r_e are constants which are determined by the material type. Graphs for various Morse potentials of different materials and for various values of D, α, and r_e are shown in Figures 5 (a) -(g).

It is important to note the behavior of the above potentials. For hard materials (such as tungsten) the potential curve is deeper and more sharply curved than for softer materials (such as iron). This is explained by the general behavior of hard vs. soft materials:

Hard Material:

- Difficult to move atom from equilibrium (and thus a sharply curved/deep potential well)
- Once bond energy is overcome tends to have brittle fracture (and thus the

sharper curvature/limited extent of significant potential)

Soft Material:

- Easier to move atom from equilibrium (potential well is more shallow)
- Tends to be more ductile in nature (thus the gentler curvature/wider extent of significant potential)

4.4 Molecular Dynamics using Morse potentials

The following is a step by step description of a basic molecular dynamics simulation procedure that uses Morse potentials.

1) Set up the coordinates of the atoms in Å. Usually in terms of an x, y, and z coordinate. The unit typically used is Å.

2) Define atom potential function to be used. Morse potentials need to be generated for each like (same material to same material) and unlike material possibility. Basically if there are N materials there will be N(N-1)/2 potential energy curves necessary.

3) Relax the atoms. Basically this procedure iterates on the set of atoms until they reach at least a local minimum in potential energy. This is necessary because quite often the atoms are set up in a "perfect" BCC or like matrix, which will therefore have high energy points at the corners and edges. Relaxing the atoms lets them "settle" to their lower energy states. Since the typical order of magnitude of an MD simulation in terms of time is roughly a few picoseconds or less, this "relaxation" period is very short indeed, and thus not able to completely relax the matrix. Thus it is important to try and set up initial coordinates of atoms to a reasonably low energy state to begin with.

4) Perform a bond check. This checks all atoms with each other to see if they are close enough to "bond". Usually a parameter known as a "cutoff radius" is associated with each potential type. This is the radius within which two atoms are considered "bonded". Beyond this radius the potential is negligible.

5) Using the bond list created, perform a Runge-Kutta or like procedure to implement the effects of the bonds.

6) Return to step 4 until desired

Some special considerations are required in a simulation of this type. These are described below:

Boundary, peripheral, and moving atoms:

Since molecular dynamics is so computationally intensive, any way to reduce the atom set required helps to greatly reduce simulation time. Take for example, a typical cutting simulation. Usually the simulation runs for a time, with the areas of interest near the cutting tip and edges as well as to a certain depth into the workpiece. Thus usually an atom set like that in Figure 6 is appropriate. However, the "real" workpiece would be much larger, indeed most likely several thousand times larger in all directions. In order to convey the stability that such a large workpiece would possess, the concept of boundary atoms are employed. The idea is to place a layer of boundary atoms at the extreme edges of the atom set, beyond which is implied to be a large amount of similar atoms. Thus, in Figure 7, it is implied that the workpiece is much larger below and to the left of the workpiece atom set, and the tool is much larger above and to the right of its atom set. These boundary atoms are not allowed to be affected by potentials, but they do exert their effect upon non-boundary atoms. Quite often a set of atoms is contained of ALL boundary atoms. This is done when it is desirable to have an "infinitely hard" tool or similar concept, as boundary atoms are not allowed to deform. Another type of special atom employed is the peripheral atoms. These atoms are clustered near the boundary atoms. Their function is to simulate the dissipation of heat out into the much bigger set of

atoms beyond the boundary atoms. This is typically done by "resetting" the atoms' velocities every so often to a generally lower value, thereby simulating cooling. Moving atoms are a term used to describe non–boundary atoms. Note that peripheral atoms are moving atoms as well. Figure 8 illustrates the boundary-peripheral atom concept for a set of atoms considered to be much larger to the left and below. Notice how the peripheral atom layer adjoins the boundary atoms.

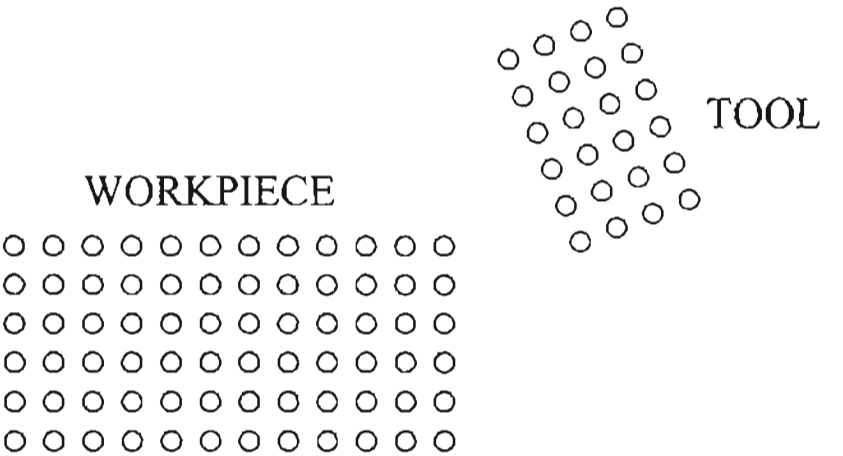


Fig. 6 : Typical MD cutting simulation atom set

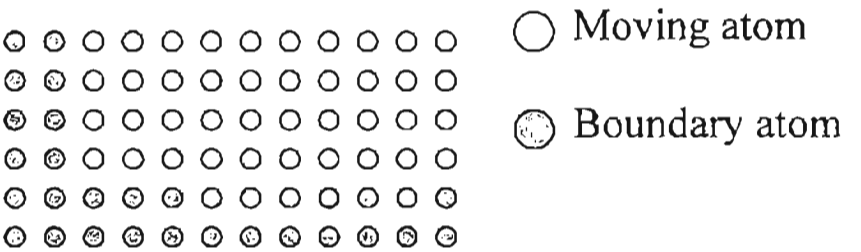


Fig. 7 : Illustration of boundary atoms

How to move a set of atoms:

Often, it is desirable (especially in cutting simulations) to induce a velocity in a certain subset of atoms. Exactly how this is done is dependent upon a few things. Most importantly, a decision has to be made about whether a simulation of constant velocity, constant force, or a different situation is desired. The most basic method is to use constant velocity. This method assumes that the forces encountered are instantly overcome by whatever is driving the tool boundary atoms (such as a high powered lathe). Constant force is not very difficult to implement a well. No matter what movement scheme is decided upon, the basic procedure is to move the boundary atoms and let the rest of the atoms in the set react to that. If a set of atoms is considered to be moving before the simulation begins, then all atoms need to be given an initial velocity to avoid the effects of instantly "jerking" the atoms into motion. Movement steps are typically very small, the exact magnitude of which will be dependent upon desired velocity and the time step used. Too large of a motion per time step can induce problems with accuracy and "overlap", where an atom moves too close or even past another atom due to large time step jumps.

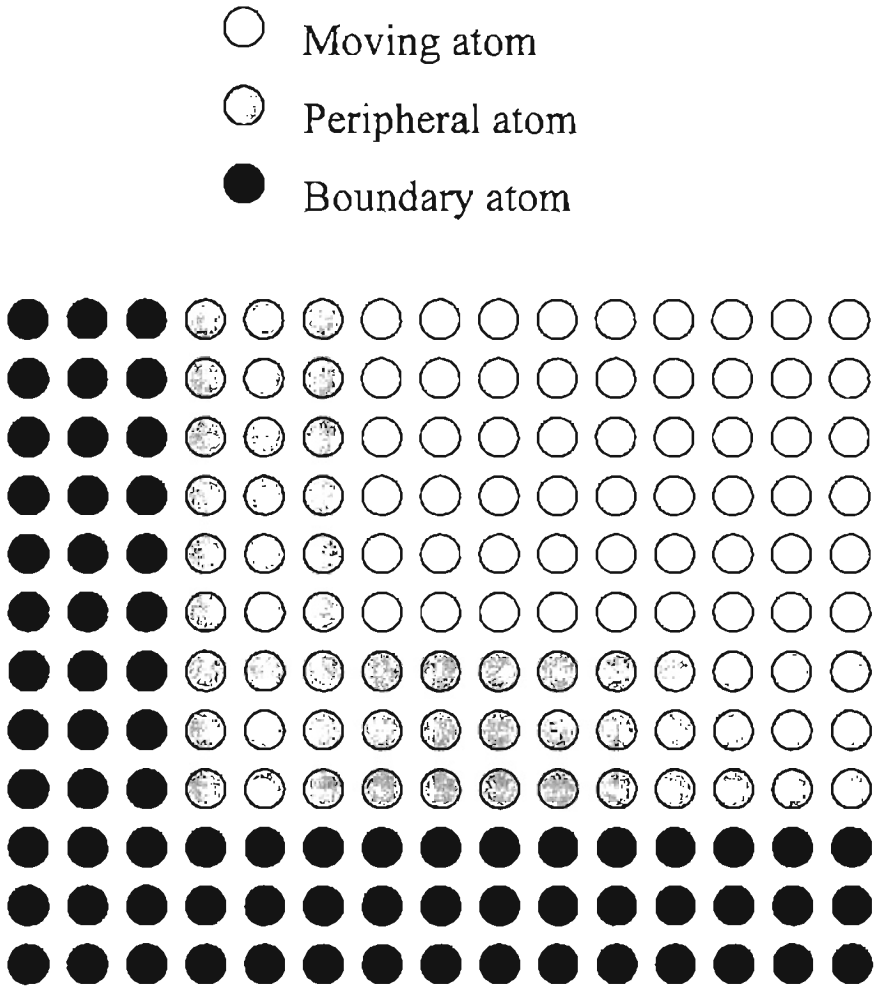


Fig 8 : Diagram of moving, peripheral, and boundary atoms

CHAPTER 5

OPTIMIZATION OF MD SIMULATION

When performing molecular dynamics calculations with the basic procedure outlined in Chapter 4, quite often the most time consuming step is the calculation of which atoms are "bonded" each time step. The standard method for doing this is to use something like the following algorithm:

For N atoms: Let atomA=1 to N-1

Let atomB=(atomA+1) to N)

Check atomA and atomB for a bond

This checks all possible pairs of atoms for a bond. The amount of checks required for N atoms is N(N-1)/2. A small workpiece of size roughly 20 nm x 20 nm x 20 nm contains on the order of 250,000 atoms or more. This translates into almost 32 billion possible bonds. It is easy to see why this step can be so time consuming. Any way in which to limit these checks would have a great impact upon performance. A new method which drastically reduces the time for this step is described below.

5.1 Cell Method Concept

The general procedure for calculating bonds results in so many checks for mainly one reason, namely, nothing is known about the position of atoms other than their actual coordinates. At first consideration, this might seem like enough information. However, the fact that every atom in the simulation is likely to move at least minimally per time step means that the bond set has to be regenerated each time step. With just atom coordinates, there is no way to tell if an atom has moved outside of or inside of another atom's "cut-off radius" without actually checking the distance between them.

Thus, having data sorted by its relative position in simulation space is desirable. Dividing the simulation space into small regions (hereby dubbed "cells") and then only comparing atoms in those adjacent regions to one another results in a large reduction in the amount of calculations required. Several implementations of this approach have been used in MD simulations (Allen and Tildesley 1991)

The "Cell Method," to be described in the following, provides much more information than just an atom's coordinates. It allows instant access to an atom's general position within the simulation space, and also access to its immediate neighbors. This results in a massive speed increase, as now an atom is only compared with its immediate neighbors. The method in detail is described below.

Step 1: Decide on a "cell size"

The cell method derives its name from a basic data storage scheme that it uses. It divides the atom space up into cubic "cells" of a predetermined size, such as in Figure 9. The idea is to include as much of the atom set as possible during the entire simulation in one large box of "cells". The size of the cells selected is usually a tradeoff between two factors: number of atoms per cell and total number of cells generated (the reasons for this will become apparent below). A cell size of about twice the maximum cut-off radius of any atom pair in the simulation is typically a good choice.

Step 2: Insert the atoms into the cells and associate a cell with an atom

First clear out the cell data structures. Then just step through the atom set and determine which cells they are in. When the cell has been determined for an atom, store that cell in some variable associated with that atom as well as the atom inside some data structure associated with that cell. The desired result is two sets of data: one that will return which cell an atom is in, and another that will return which atoms are in a particular cell. Pseudocode that does this is as follows:

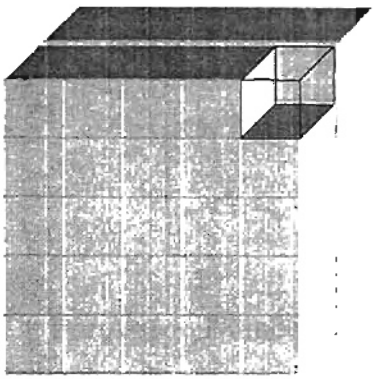


Fig 9 : A 2 X 5 X 5 set of cells

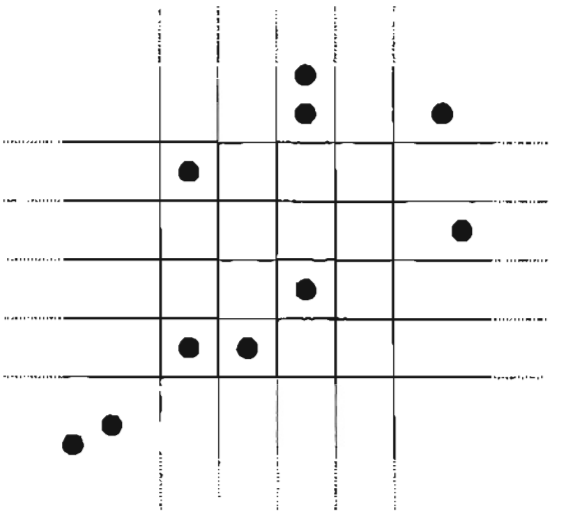
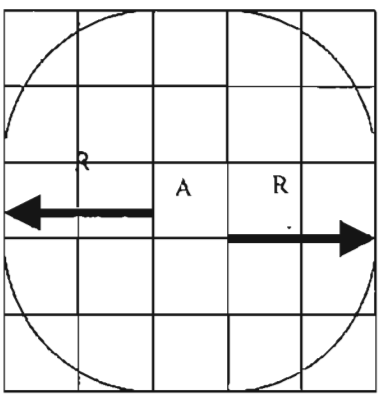


Fig. 10 : A 2d set of cells, with the exterior cells extending to infinity



A = Cell containing atom in question

R = "cutoff" radius = 2 cell widths

Fig 11 : Illustration of number of necessary cells to check for bonds

For J=1 to NumberofCells

Clear AtomsinCell[J]

For A=1 to NumberofAtoms

C= which cell atom A lies within

CellofAtom[A]=C

Add A to AtomsinCell[C] data structure

For atoms outside the "cell set" there are additional "infinite cells" to place them in. These are shown in Figure 10. The bounding cells effectively stretch to infinity.

Step 3: Go through atom set and determine bonds

This is where the benefits of the cell method should become clear. To generate the bond set, the following procedure is followed:

1) Go through the atom set

2) For each atom, determine which cell it is in (already known from above)

3) Check for bonds with the atoms in the same cell or within a certain number of adjacent cells

What is this "certain number"? Figure 11 displays how this is determined. The number of adjacent cells checked needs to fully cover all positions within the cut-off radius of the current atom. This is why choosing the cell size based upon the maximum cut-off radius encountered is a good practice.

5.2 Comparison of Cell Method and Standard Method

The cell method becomes much faster than the "standard method" for many reasons. The basic costs of both methods are outlined below:

Standard Method: For N atoms, N(N-1)/2 distance calculations

Cell Method: For C cells and N atoms,

a) C cell structure clears

b) N checks to see which cell atom an atom is in

c) N insertions into "CellofAtom" data structure

d) N insertions into "AtomsinCell" data structure

e) roughly A*B*N distance checks, where A = average number of atoms per cell, B = number of cells necessary to be checked per atom

Even though distance checks and data insertions do not typically have the same time cost in a computer, it is not unreasonable to take the above lists and generalize that the timing for the standard method is N(N-1)/2 and the cell method is (C/N+3+A*B)N. It can be seen that the timing for the standard method is exponential with N, whereas the cell method is roughly linear with N. The importance of cell size and number of cells can be seen in the timing equation for the cell method. Decreasing cell size but still covering the same volume increases the number of cells C, decreases A and increases B. Larger cells (and less of them) do the opposite.

The cell method was put to the test in several cutting simulations to get actual timings. The computer used was a DEC 500 MHz Alpha running Digital Unix. Three versions of the simulations were performed with each atom set: one with the standard bond generation method, the second with the cell method, and the third with no bond generation (skipping that step). The no bond generation simulation was run to give an idea of how much time the other steps besides bond generation required during

simulation.

Table 1 : Timings of various bond generation methods. The number of cells used were 40x40x40. The cell size was 3.0 Å, with the maximum cut-off radius being less than 6.0 Å.

No. of Atoms	Cutting Distance	No cutoff timing (Hr: Min: Sec)	Cell method timing	Standard method timing
2930	50 Å	00:07:25	00:11:27	00:19:41
6856	50 Å	00:42:50	00:57:36	03:25:08
12056	100 Å	03:10:37	03:55:50	25:38:21

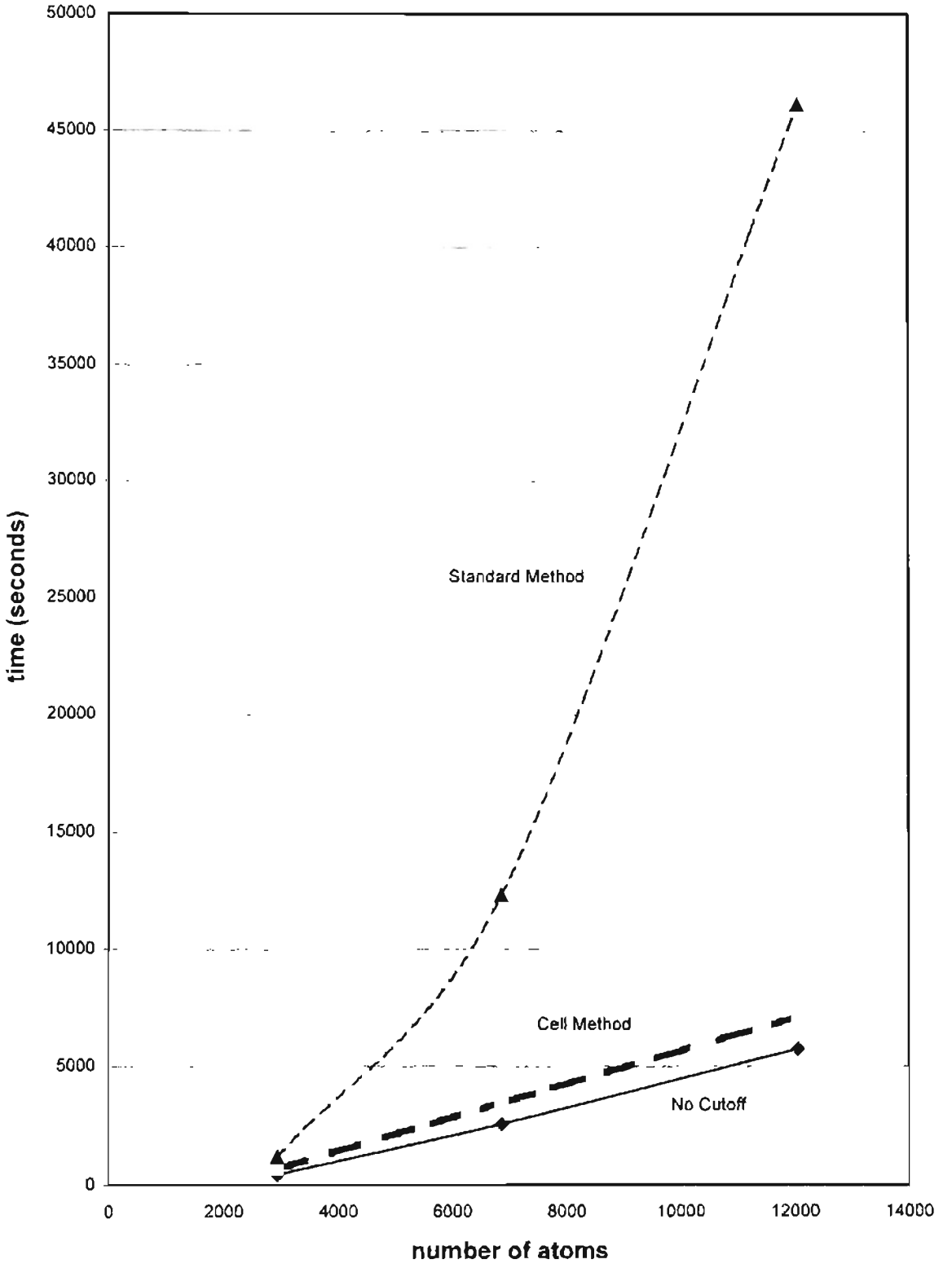


Figure 12 : Timings of various bond generation methods

These results are summarized in Figure 12. It clearly displays the cell method's massive advantage over the standard method, especially as the atom count increases. The cell method's timing increases only slightly faster than the equivalent simulation with no cut-off, while the standard method's timing increases exponentially. Note that the difference between both the cell method vs. no cut-off and the standard method vs. no cut-off is how much time the cut-off routine takes in the respective methods. As one can see, when the cell method is utilized the cut-off timing required is small compared to the rest of the simulation (Runge-Kutta, etc.) while in the standard method the cut-off timing dominates the computations.

5.3 Integration of Cell Method with Other Optimizations

The cell method greatly reduces the time required for the cut-off routine in a typical molecular dynamics simulation of a large number of atoms. The cell method also lends itself well, however, to integration with other optimization techniques. One of the most relevant (to cutting simulations) and useful optimizations is the "area-limiting" concept. Basically the idea is to limit which atoms are operated on out of the atom set to those which would experience non-negligible effects. As applied to a cutting simulation, this typically involves atoms close to the cutting tip and those in the chip. Due to the fact that this method reduces atom count, the timings will decrease for the cut-off routine. Figure 13 displays the basic concept behind the area limiting principle.

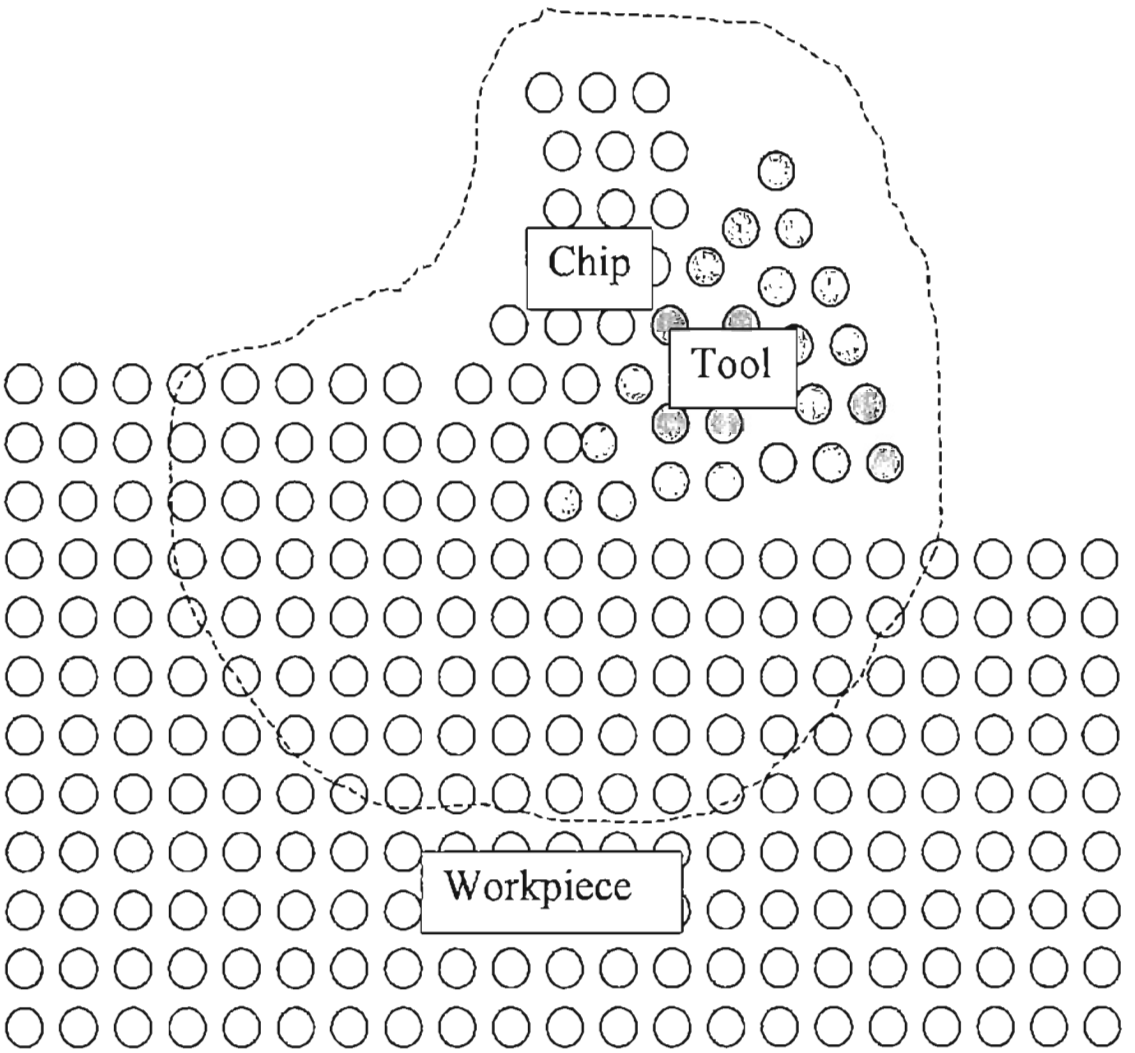


Figure 13 : Example of area limiting

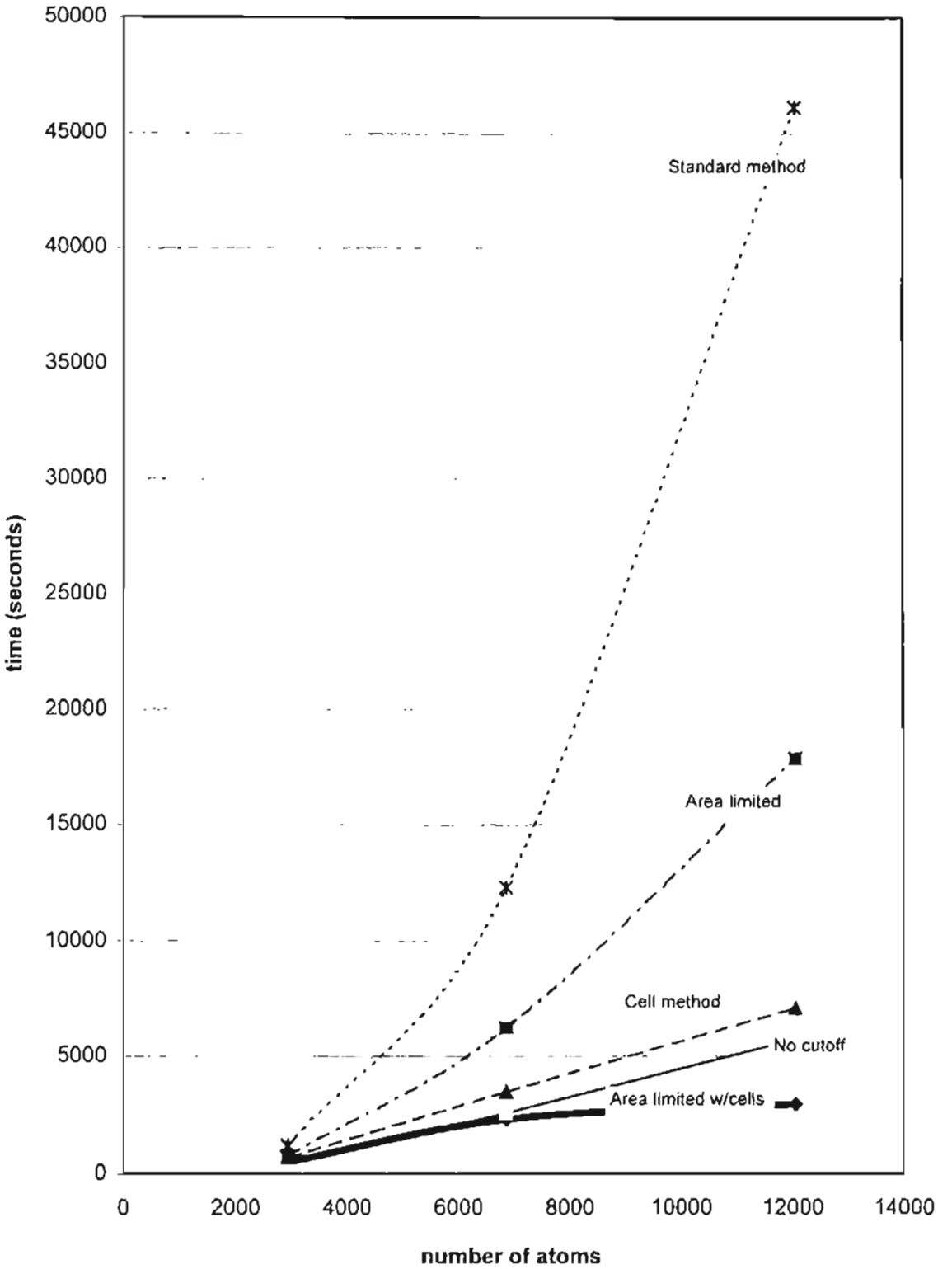


Figure 14 : Timings of various bond generation methods

Some special considerations to take into account when performing such an "area-limiting" approach is that the fringes of the area considered is likely to have more atoms beyond it. Therefore it is common practice to make atoms which are not in the chip and near the edges boundary atoms until they are well within the area.

Two versions of molecular dynamics simulations were performed to show the effect of an area restricted only method and an area restricted with the cell method added on. These are contrasted with the earlier standard, no cut-off, and cell method only runs.

Table 2 : Timings for various bond generation methods including area limiting.

Number of Atoms	Cutting Distance	Method	Time (Hr:Min:Sec)
2930	50 Å	No cutoff	00:07:25
		Area limited w/cells	00:08:23
		Cell method	00:11:27
		Area limited	00:12:30

		Standard method	00:19:41
6856	50 Å	No cutoff	00:42:50
		Area limited w/cells	00:38:20
		Cell method	00:57:36
		Area limited	01:43:42
		Standard method	03:25:08
12056	100 Å	No cutoff	03:10:37
		Area limited w/cells	01:37:46
		Cell method	03:55:50
		Area limited	04:58:24
		Standard method	25:38:21

The above table is summarized in Figure 14. As can be seen, the area limiting alone tends to perform between the standard method and the cell method over the atom set range tested. Area limiting with the cell method begins to greatly outperform, even the no cut-off procedure. This is due to the fact that area limiting reduces the total instantaneous atom set size, whereas no cut-off simply skips the cut-off procedure but performs other operations on the full atom set (such as Runge-Kutta etc.).

5.4 Computational Issues

The time saving advantages of the cell method (as well as the area limiting method) are indeed attractive. However, the question of loss of accuracy arises. For an optimization like area-limiting, the main issue is how far away from the tool tip the area considered needs to be in order to capture all non-negligible interactions. If the chosen area does not include all of the non-negligible interactions, the results will differ from the standard method.

The cell method on the surface seems to not introduce any error. It simply is a much faster way of generating the same set of bonds within an atom set. The order in which it generates the bonds differs from the standard method. Since it is the same set of bonds (just in a different order) this would also seem to not create any differences. However, the way in which computers perform basic addition operations influences the results. Identifying the sources of such errors and reducing them are of importance to MD simulations, as the amount of computations performed is typically very high. Therefore, even a small amount of error per calculation has the potential to increase to a significant amount of error over many millions of calculations. Any method to reduce the amount of error accumulated would be of value.

5.5 Computers and Floating Point Math

Computers are basically digital devices in that they store numbers in a digital format. What this means is that there are discrete "steps" between successive numbers that can be represented in a computer. Therefore, some numbers cannot be represented exactly, but must be rounded. This is one source of mathematical error. Perhaps more important is the fact that most computers can only store so many digits of precision per number. Most computers store floating point numbers according to the IEEE standard, which includes a mantissa and an exponent. The range of the mantissa defines the precision of the number. This is a set number of digits in most cases (based upon the number of bits available for the mantissa in the computer's representation of floating point numbers). The fact that the precision is set can create some errors when performing basic arithmetic operations on large sets of numbers. To illustrate this behavior, some examples are provided below.

Results of adding an array X:

X is an array filled with the following:

For n=1 to 1,000,000 : angle = 1.5(n-1)/1,000,000 radians , $X(n) = \sin^2(\text{angle})$

Thus X basically contains all positive numbers increasing in magnitude from 0.0 to about 1.0. The array was added in three different fashions : randomly, smallest value to largest value, and largest value to smallest value. The following results were obtained:

Randomly:	476 479 . 501 158 585 770
Smallest to Largest:	476 479 . 501 158 594 620
Largest to Smallest:	476 479 . 501 158 581 000

The results are in keeping with the fact that adding numbers smallest to largest minimizes error. Adding the numbers smallest to the largest gave the largest answer. This is because the true sum of the smallest numbers was retained more precisely than in the other two cases. Adding the numbers largest to smallest resulted in the lowest answer. This is because once the small numbers were reached, the sum to that point had already become large, therefore the small numbers' rightmost digits become truncated (since the computer will only consider a certain precision). The random result was between these two extremes.

To contrast this example, a large set of numbers with the same precision range were added in the same fashion. The idea here was to maintain the precision not only in the numbers being added but in the sum as well. Therefore the following scheme was chosen:

For n=1 to 100,000 ;

a = 0.00000000001 ; b = a+1 ; D(1) = a; D(n) = d(n-1)+a;

The array D was added to b randomly, smallest to largest, and largest to smallest. In each case, the same result was obtained:

Randomly: 1 . 050 000 500 009 975 20

Smallest to Largest: 1 . 050 000 500 009 975 20

Largest to Smallest: 1 . 050 000 500 009 975 20

The same result was obtained because as the sum was generated, the precision of the next number to be added matched that of the current sum. Starting out with the sum at b ensured this. Thus, no losses were incurred due to the number being too small as compared to the current sum.

Therefore, if one can ensure that both the numbers being added and the sum are always equal, in terms of decimal place precision, then minimal error is encountered with any addition order. However, this situation is rare. In terms of application to molecular dynamics, the most commonly added values are potential calculations, i.e. adding up the

potentials experienced by each bond. It would be possible to define a precision range that is "middle of the road" for the potential function but this would lose some detail. For example, suppose that a Morse potential varies on value from 1.5689×10^{-2} to 6.3569×10^{2}. Also suppose that it is known that the sums of the potentials will never be more than the value 1.0000×10^{3}, and that the computer used can represent 5 digits of precision. Thus the sum could be initialized to 1.0000×10^{3}, and then the potentials added to it in any order with the result being the same. However, the lowest potential value of 1.5689×10^{-2} and indeed all numbers below 0.1 are now effectively zero, as their precision lie out of the 5 digit range of the sum. So accuracy of calculation is lost for a gain in repeatability. In most cases for molecular dynamics retaining accuracy would be preferable.

The effects of out of order addition behavior upon molecular dynamics simulations are shown below. Four methods are examined: (1) the standard method, (2) the cut-off method, (3) the area limiting method, and (4) the area liming method combined with the cell method. In each case, the calculations will differ due to varying reasons. When the cell method is utilized, the bond list generated will be out of order as compared to the standard method. When area limiting is utilized, only some atoms are operated upon out of the entire atom set, thus inducing differences as compared to the standard method.

Figures 15 (a)- (d) display the atom positions for a molecular dynamics simulation

using the four methods examined. From these graphs, the differences are apparent but the basic behavior is similar. One point that should be considered is that if an atom is at a "neutral point", i.e. it is at equilibrium, then a small calculation difference could send it right, up, left, or down. These are the primary reasons for the visual differences seen in the above graphs. A better way to quantify the differences is to analyze the force history on the tool. Since knowledge of the forces experienced by a tool during a cutting operation is of great engineering interest, this is a good way to determine if the induced error due to out of order/less than full atom set calculations is acceptable.

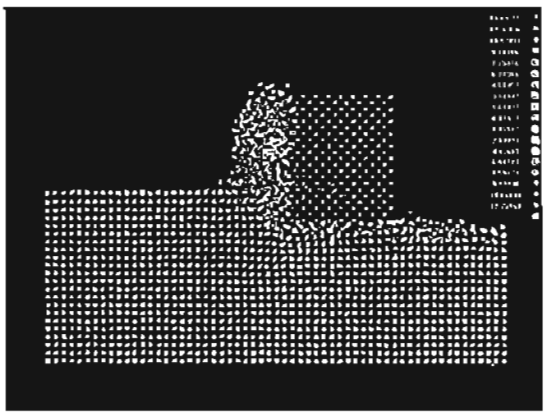


Figure 15a : Standard method

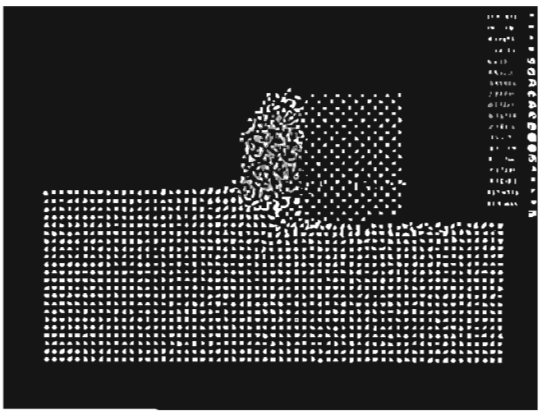


Figure 15b : Area limited

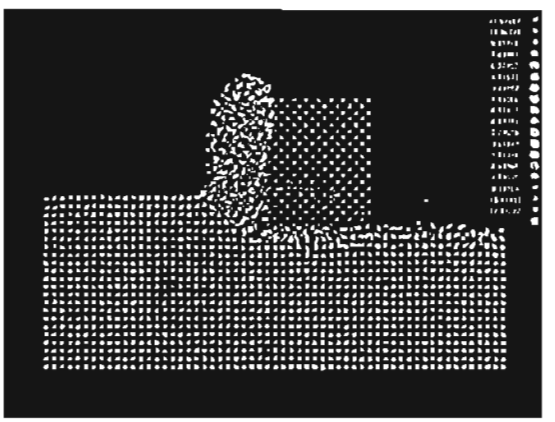


Figure 15c : Cell method

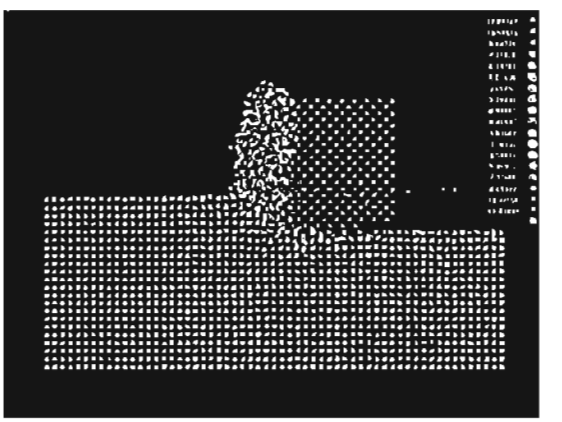


Figure 15d : Area limited w/cells

Figures 16 (a)- (d) display the forces for the same four simulations. From them, several behavioral patterns can be seen. All of the four force plots are extremely similar near the start of the simulation (the right of the plots, since the tool moved right to left). This is because any error that is building up due to out of order calculations and/or area limiting has not become significant. However, as the simulation progresses, the four plots begin to diverge. The cell method only plot differs slightly from but follows the same trend as the standard method plot, while the area limited only plot begins to differ more prominently from the standard plot as the simulation progresses right to left.

The similarity of the cell method to the standard method implies that the amount of error incurred to this point due to out of order addition is not yet significant. Also notice that the area limiting with cell method is very similar to the area limited only plot. Once again, this implies that the out of order addition has not manifested significant error yet.

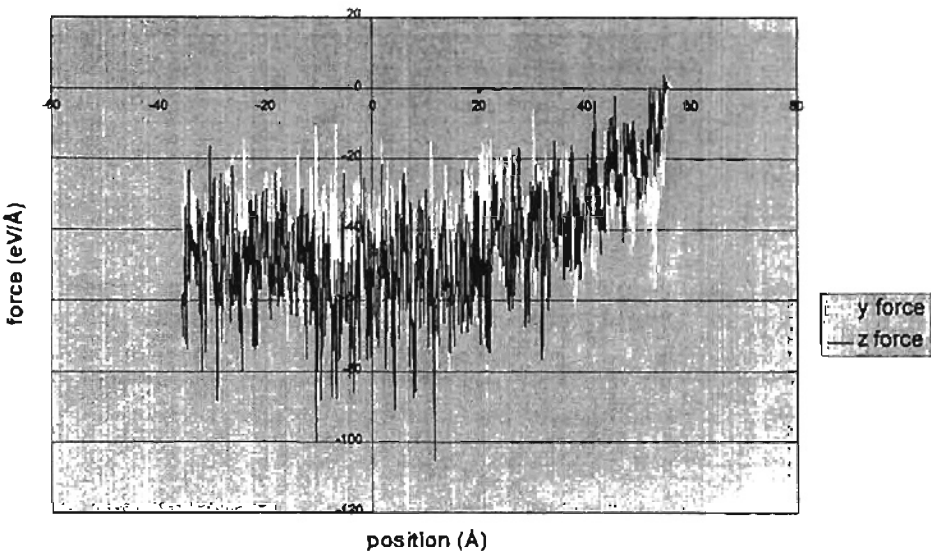


Figure 16a : Standard method

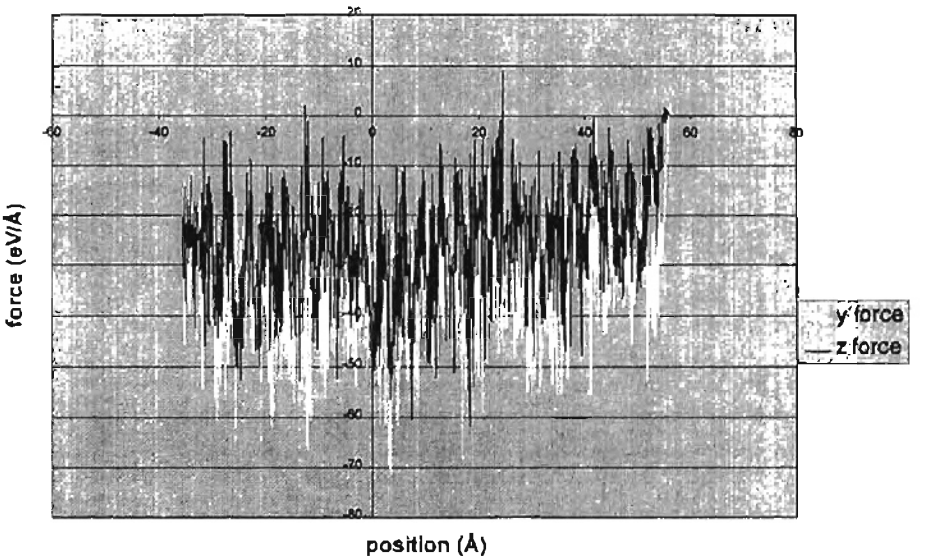


Figure 16b : Area limited

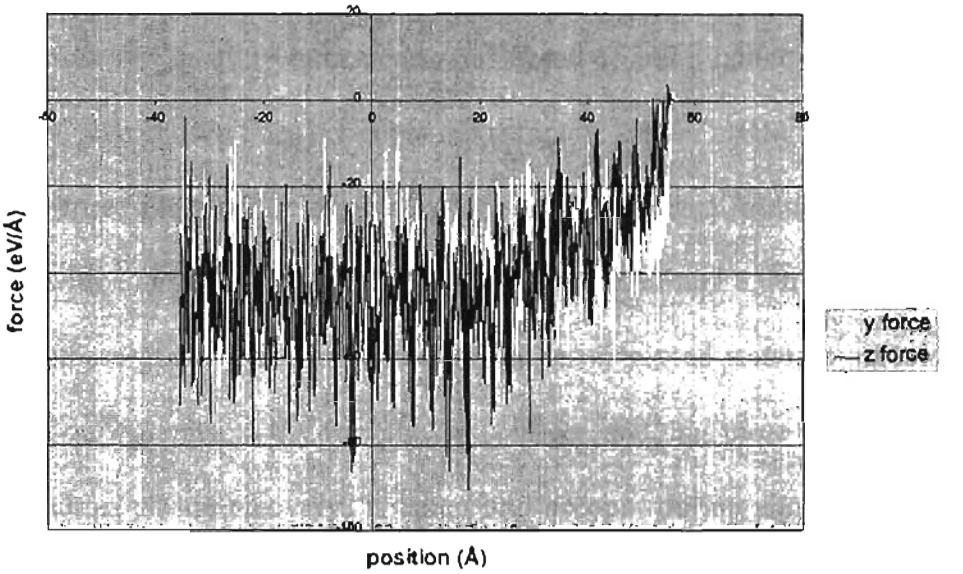


Figure 16c : Cell method

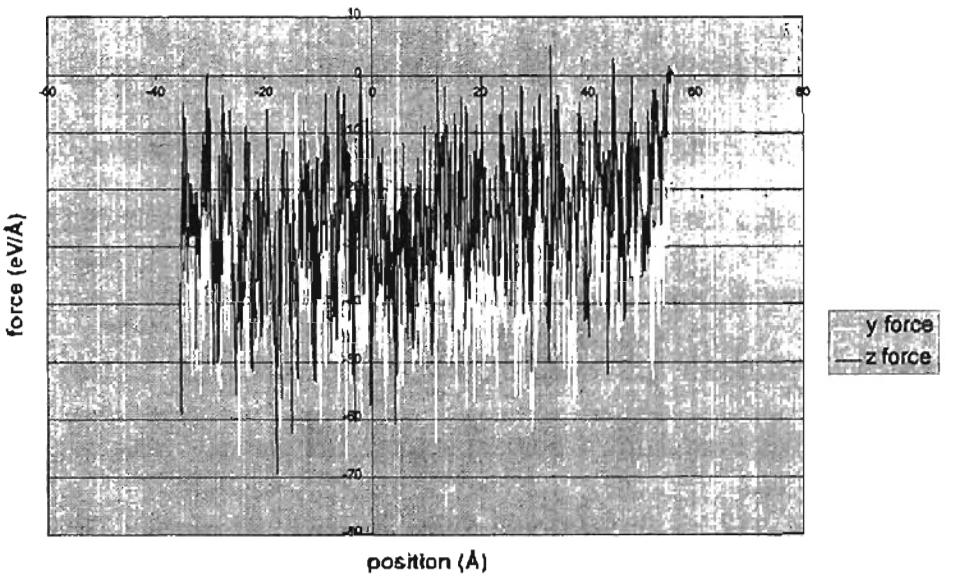


Figure 16d : Area Limited with Cell Method

The area limited only method is producing pronounced error due to the fact that the area chosen was small (9 lattice constants ahead of and 3 lattice constants behind the tool tip). The area chosen does not cover the complete area of elastic recovery. Figures 17 (a)- (b) are force plots of the same simulation run with area limiting/area limiting and cells, only with a larger area (12 lattice constants ahead of and 8 lattice constant behind). Notice that these graphs look more like the earlier displayed standard method only and cell method only graphs, as the area chosen is larger, and thus more of the non-negligible activity (such as elastic recovery) is taken into account during simulation. Notice once again that the cell method does little to alter the results here as well.

It is important to reiterate the fact that the standard method is not "more correct" than the cell method. The standard method generates bonds by following a certain algorithm. The values of the potentials of those bonds will not coincide in terms of what order they are in (smallest to largest order, etc.) The only way to guarantee that is to sort them. The goal of the above graphs was to show that the differences between two methods (namely the standard and the cell methods) which have different addition orders are much lesser than when important atom bonds (as in the case with the smaller area limiting simulation) are omitted.

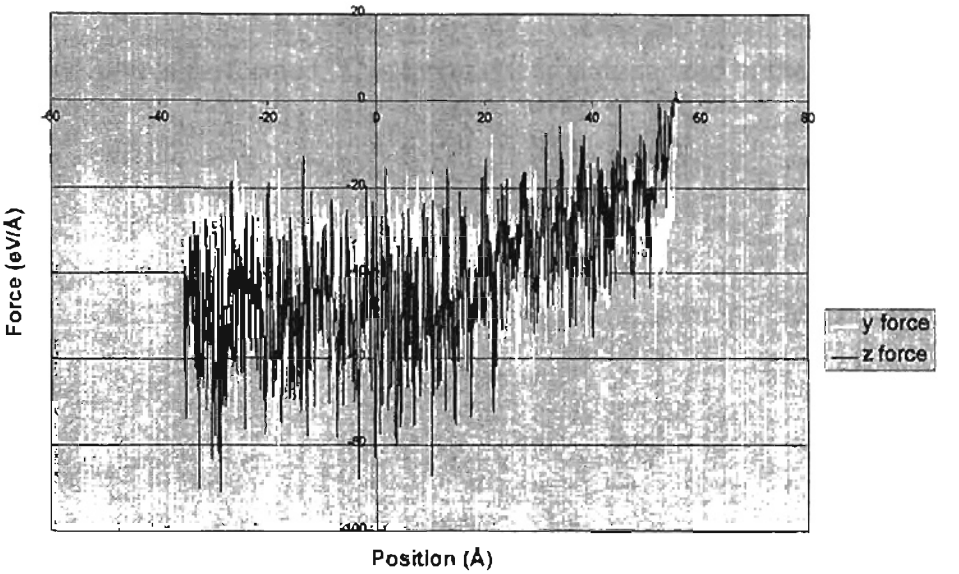


Figure 17a : Area limited

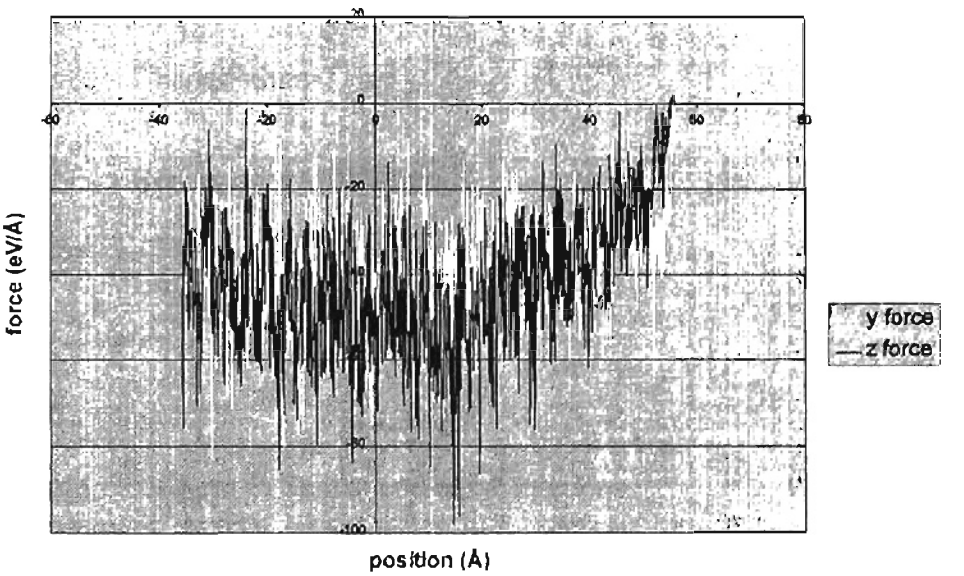


Figure 17b : Area Limited with Cell Method

A fast method that could be used to minimize error is to continuously update an error term as addition is performed. This technique is summarized in the following ::

A[n] is the array of numbers to be added

For b=1 to n :

Temp=Sum

Sum=Sum+A[b]

Change=Sum-Temp

Remainder=A[b]-Change

ErrorTerm=ErrorTerm+Remainder

Add ErrorTerm to Sum at end of calculations

In this algorithm, ErrorTerm will accumulate most of the portions of A[n] which are truncated during addition to Sum. Some losses still can occur, if the precision range of A[n] varies widely. The cost of this approach is not too excessive, basically just 5 steps

in stead of one for addition. The precision range of calculations can be as much as doubled by this procedure. This method was applied towards two MD simulations to evaluate how much floating-point error was present. The simulations conducted were as follows:

10,110 : 10 was simulation run using conventional method, 110 with error correction

12,112: 12 was simulation run using conventional method, 112 with error correction

Figures 18a-18d display force graphs for these simulations.

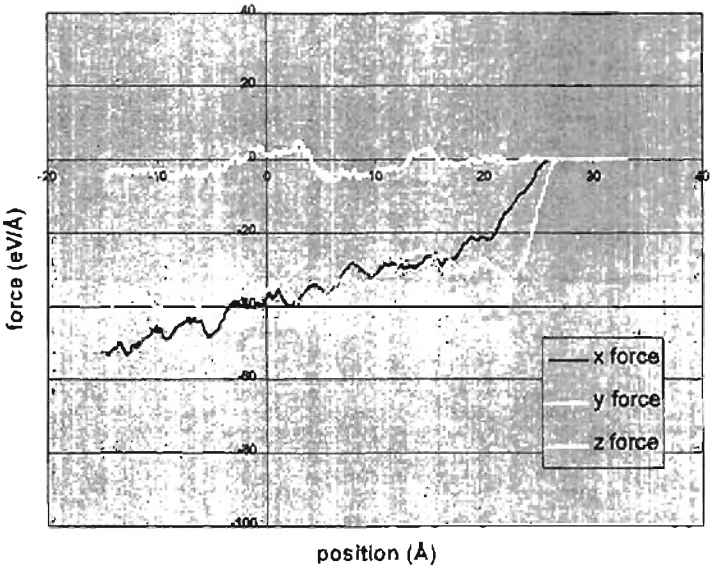


Figure 18a : No error correction

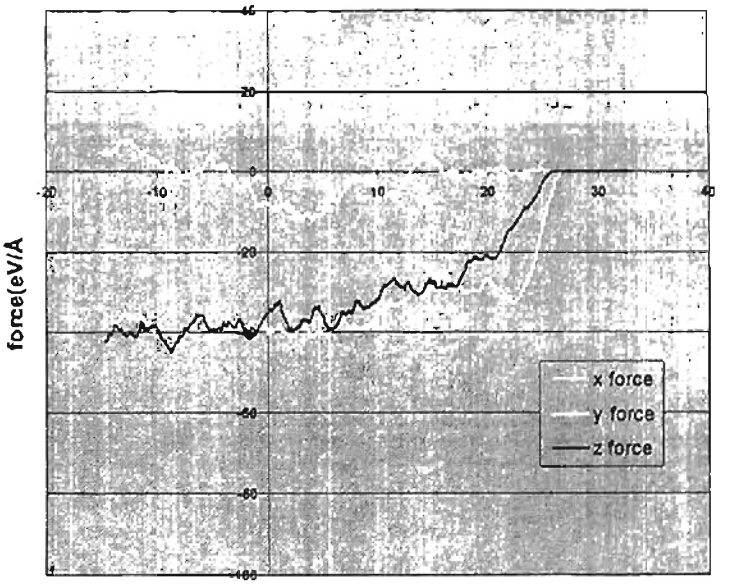


Figure 18b : Error corrected

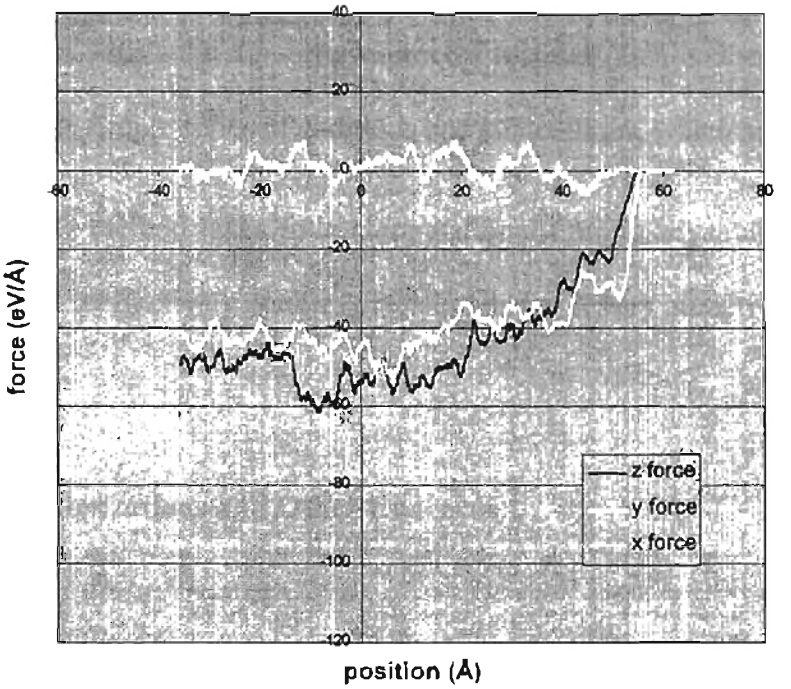


Figure 18c : No error correction

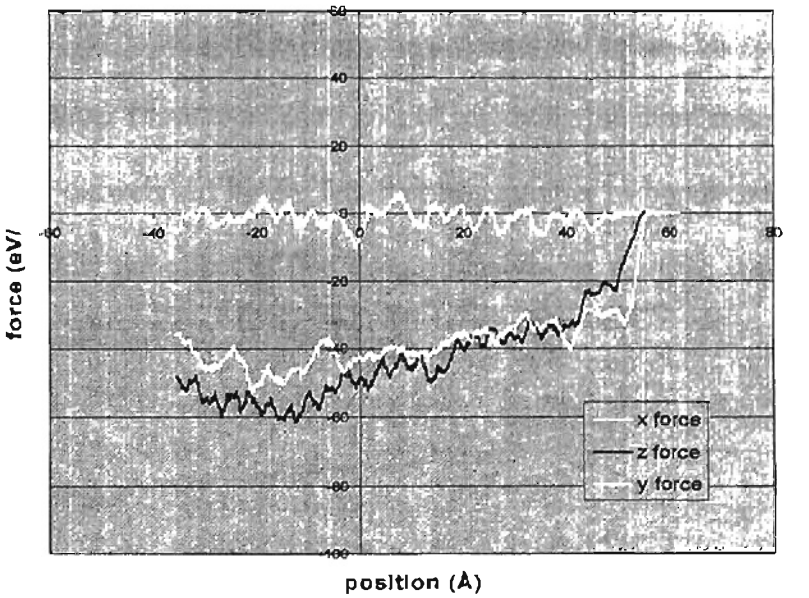


Figure 18d : Error corrected

In both of the above comparisons computational time was less than 5% more for the error corrected method versus the non-error corrected method. This is due to the fact that the error correction is simple series of additions and subtractions, which are relatively quick.

The fact that this method takes little computational time is advantageous, and coupled with the fact that it effectively doubles precision makes it very attractive.

5.6 Further Optimizations and Other Concerns

There are a few ways in which it is possible to further optimize the cell method, although not all are desirable. The first is to practice a 'removal" concept. Basically once an atom has all of its bonds generated, remove it from the cells. This prevents bonds from being checked twice. There is an extra cost of removing the atom from the cells in terms of time, but it is outweighed by the fact that less number of bonds are checked. Table 3 and Figure 19 show the effect of this extra procedure on the time it takes to generate a bond list vs. the standard and cell methods:

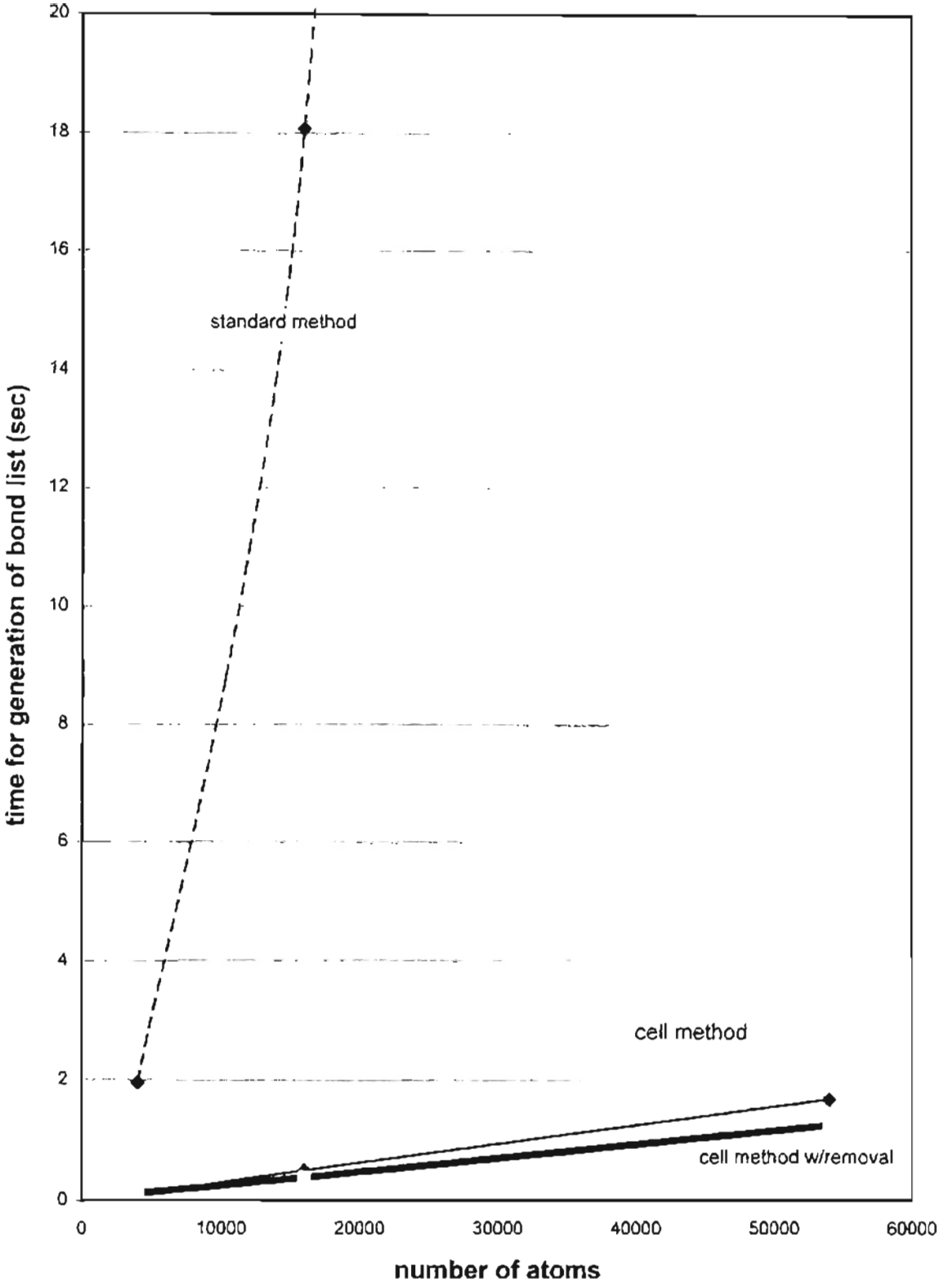


Figure 19 : contrast of timings for generation of bond list

Table 3 : Comparison of removal to non-removal timings

Number of atoms	Method	Time (seconds)	Number of bonds checked
4000	cell method	0.0833	254,889
	cell method w/removal	0.1167	135,133
	standard method	0.4666	1,954,150
16000	cell method	0.5000	2,506,508
	cell method w/removal	0.3833	1,276,494
	standard method	18.826	127,272,600
54000	cell method	1.6833	9,912,720
	cell method w/removal	1.2500	5,034,094
	standard method	183.3427	1,454,329,350

The removal method is clearly about 25% or so faster than the cell method

without removal. However, certain issues need to be considered. First, the cells contain information that may be useful to other algorithms, and so "destroying" that information by removing the atoms during bond checking may not be desired. Second, the gain in total speed for a small to mid-sized computation by reducing the cell method's cut-off routine by 25% is minimal, as Runge-Kutta and other operations now dominate the timing. The extra code may not be desirable. Certainly for larger runs the time gained may be more significant, and in general the removal method is probably a desirable option.

As mentioned earlier, the idea of the cell method is to try and contain as many of the atoms throughout the simulation in the "interior" (non-infinite sized) cells. The effects of atoms lying outside the interior cells on calculation speed are shown in Figure 20. All of the timings were for the same atom set (20000 atoms) with one difference: "configuration". This was set up as follows:

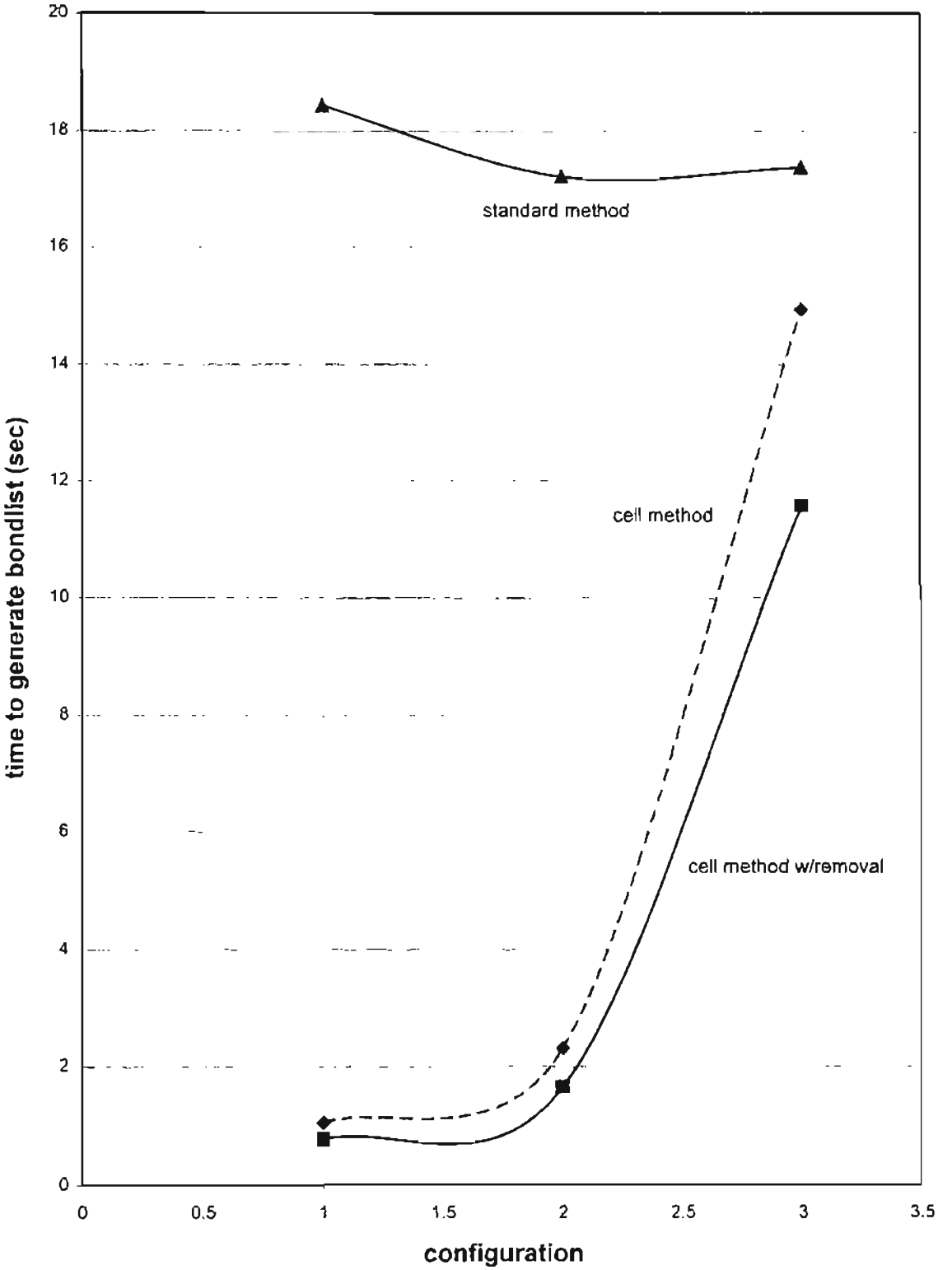


Figure 20 : Comparison of timings of bondlist generation for various configurations of atoms

Configuration 1: all atoms lie completely within the interior cells

Configuration 2: half of the atoms lie within the interior cells

Configuration 3: almost all of the atoms lie outside the interior cells

As can be seen from the above chart, as more and more of the atoms escape, the volume covered by the interior cells, the timing of the cell method cut-off routine begins to mirror that of the standard cut-off routine. This is due to the fact that the exterior cells (the cells that extend to infinity) have to check all of the atoms contained within them with each other for possible bonds, as their relative distances are unknown. All that is known is that they lie within the same cell, which is of infinite size. Thus, all of the advantages of the cell method evaporate. This is why it is important to carefully choose the cell placement if they are static to include as many of the atoms as possible during the simulation within the interior cells. This is not difficult for cutting simulations, as chip/tool/workpiece behavior in terms of general position is somewhat easy to predict. For very large or long simulations, it might be desirable to alter the placement of the cells during the simulation to "keep up" with the atom positions. This should be done without compromising ease of cell calculation (i.e. keep a box shape of cubic sized cells). A simple example would be to move the center of the cells with the center of gravity of the simulation.

5.7 Recommendations

The cell method greatly speeds up standard Morse potential based cutting simulations. The cut-off routine timing becomes less significant than the rest of the calculations (over the atom set size analyzed). Whereas before, the increase in cut-off timing was basically second order with number of atoms, the increase now was more linear in aspect. The typical gain in speed for an average sized simulation was 4-5 times. Since the cell method is simply a quicker way of generating the correct bond list, it merges well with other optimizations (such as area limiting). Issues arise due to the fact that the bond list created with the cell method will likely be the same but "out of order" compared to another bond list generation method. This is mainly an effect of the way in which computers perform floating point arithmetic. For small to mid-sized simulations, the differences induced by out of order bonds seemed to be minimal, but could manifest itself to an undesirable degree in longer or larger computations. A simple method which is fast, efficient, and effectively doubles precision without affecting runtime could be implemented. Another slight disadvantage to the cell method is a larger memory requirements, but these are more than offset by the increase in speed.

As processors continue to get faster and cheaper the small workstation user will continue to reap the benefits of improved performance. The method presented above, as well as 95% of all software available, will continue to run better and faster on these new single processor machines. However, making the move to true parallel processing would give

incredible benefits to a molecular dynamics simulation. The "cell method" presented above would now able to be implemented in more of a hardware model than a software model. Multiple processors could serve as the cells, passing information to neighboring processors about which atoms they contain and which atoms are leaving their "cell". Ideally, a correctly programmed simulation with one processor per cell would greatly outperform the single processor version in the cut-off routine. For small to mid-sized calculations this would not be too noticeable, as it has been shown that for these sized simulations Runge-Kutta and other steps dominate once the cell method is implemented. For very large simulations the cut-off routine with the cell method could very well overtake and surpass the other calculations in terms of CPU time, and would benefit from such an approach. Very large scale true parallel processing machines are still quite expensive and usually the domain of large national laboratories and corporations.

As shown above, after implementing the cell method Runge-Kutta and other calculations dominate the timing for small to mid-sized simulations. Investigations into different ways to simultaneously evaluate the motions of the atoms could reduce these timings. Runge-Kutta has proven to be very accurate and efficient in the past. So, perhaps the avenue that needs to be followed is not choosing a completely new evaluation procedure but implementing ways in which to provide the Runge-Kutta routine with necessary information more efficiently and quickly than the current algorithm.

CHAPTER 6

Computer Graphics Techniques for Molecular Dynamics

There are basically two tools to analyze the results of an MD simulation. One is to stay strictly in the numeric realm, and look at things such as forces, average potential energies, etc. However, perhaps of most use is to display the atoms in a picture or animation. This allows for visualization of certain phenomena which data alone tends to not reveal (such as dislocation motion, etc.). This leads to a more intuitive approach to analysis and when coupled with the strict numeric behavior can greatly aid the research process. The following is a general description of how to go about such visualization, along with possible problems and improvements that could be implemented.

6.1 Basic Computer Graphics

Most computers have a region of memory (either located in the system's RAM or on a graphics card) which is designated as video memory. This memory is typically oriented as shown in Figure 21. The entire screen can be said to be a large one dimensional array. The upper left of the screen is the first element in the video memory array, whereas the lower right is the last. Memory locations increase from left to right. The amount of colors possible per pixel determines the number of bytes necessary per

pixel. For a 256 color display, one byte is required per pixel. Thus the screen can basically be thought of as a one dimensional array, and so if a desired X-, Y- screen locations are known, then the proper index into that array can be calculated. This "linear" memory orientation is much easier to deal with than in the past, when memory was often split into various planes and banks. There are still some slight deviations from this on some graphics cards, namely an arrangement known as "pitch". The pitch of a region of video memory is how many bytes of memory there are per line of resolution. Normally one would expect this would equal the horizontal resolution (in bytes) but this is not always the case. Some cards have extra data at the end of each line, due to the fact that their architecture is tuned for only certain horizontal values. This is effectively a region that is offscreen to the right, as shown in Figure 22. The onscreen surface displayed in Figure 22 is 640x480 bytes, but at the end of each line of video memory there is an additional 100 bytes that is not displayed. Thus the "pitch" of this surface is 740. However, for calculations of x, y this "pitch" must be considered.

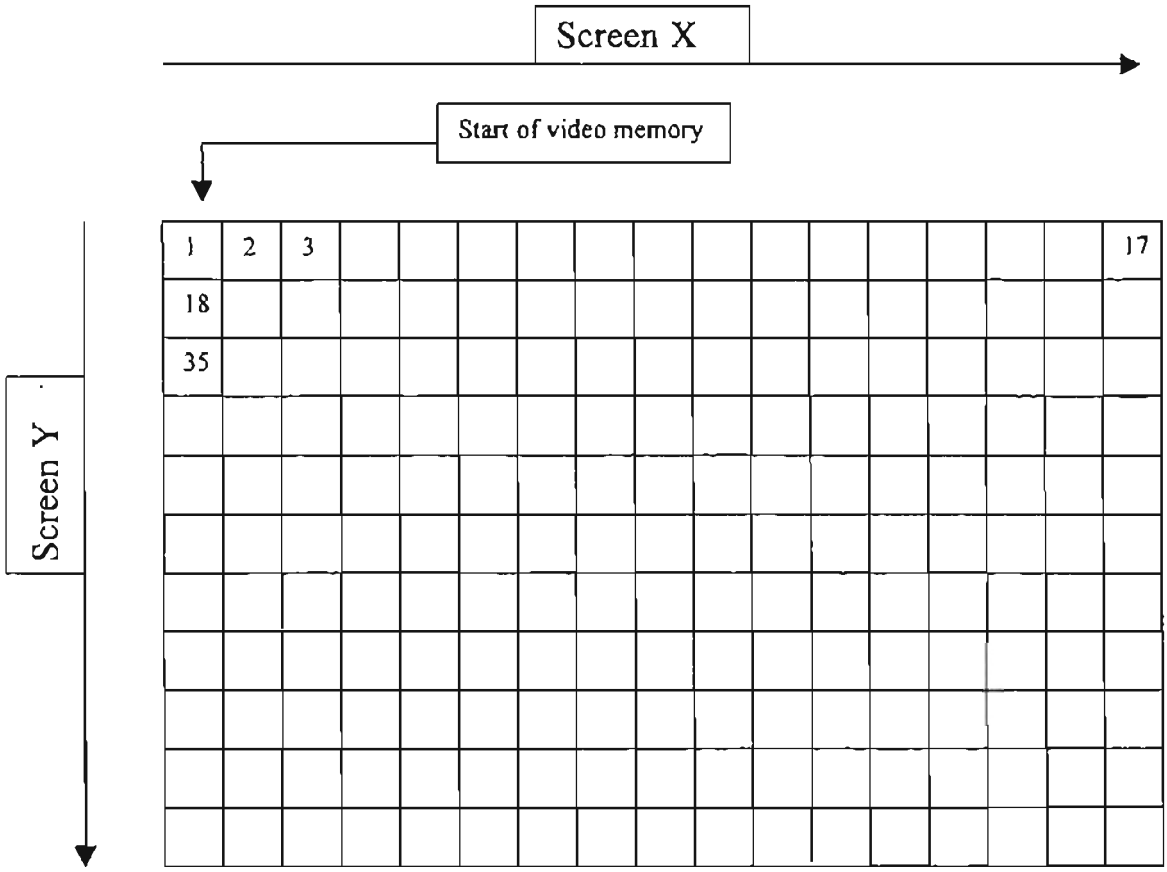


Figure 21 : Typical arrangement of video memory.

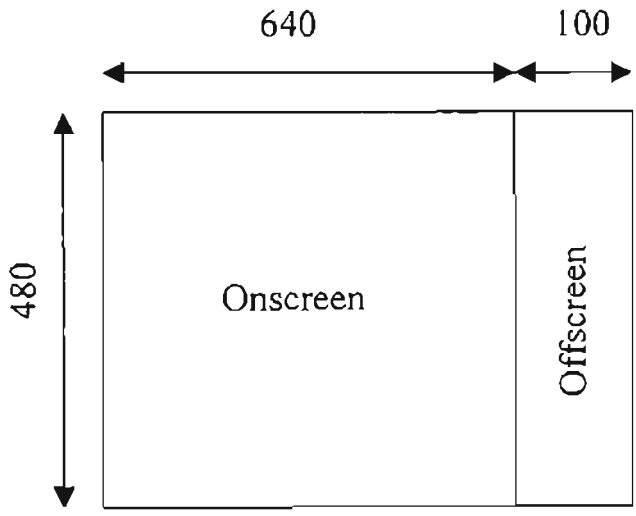


Figure 22 : Example of pitch.

Figuring out an individual pixel location is easy, and can be accomplished by the following formula:

arrayindex=Y*(pitchwidth in bytes)+X*(number of bytes per pixel)

where X and Y are the desired screen X and Y of the pixel.

Note that Y=0 is the top of the screen, and Y increases with movement to the bottom of the screen. X is zero at the left, and increases towards the right.

When drawing a picture, one could write routines which calculate the proper indices for all the pixels and set the array location to the appropriate value(s) based upon the desired color(s). Before the advent of hardware acceleration it was common practice to write routines to create lines, bitmaps, etc. completely in software. Today most graphics cards support hardware versions of these routines, which allow for a much faster performance because the CPU is relieved of much of the burden of calculation. The CPU merely has to say "copy this bitmap to this location on the screen" to the graphics card and then the card takes over. The one area where the programmer can still typically influence performance the most (and of particular interest to MD simulation) is in sorting the data efficiently.

6.2 Sorting MD data for Visualization

When drawing atoms to the screen it is usually acceptable to use a two-dimensional bitmap which resembles a 3D sphere. The issue that needs to be addressed is that of what order should they be drawn in? The answer for convex objects such as spheres is back to front. So one could sort the atom set in terms of distance from the viewer (removing any atoms that are off the screen in the horizontal and vertical directions) and then draw them in order from farthest to closest. Sorting the atoms completely can be time consuming, especially when there are many thousands of atoms that need to be drawn per frame. However, sorting completely is not required. In MD simulations, atoms rarely, if ever, move closer than 0.5 A or so to each other, due to the tremendous repulsive force present at such distances. Therefore, sorting down to the nearest 0.5 A is "good enough", as no two atoms will lie 0.5 A or less apart. A sort of this type can be tremendously faster than a traditional "complete" sort. The basic algorithm is something like:

1. Go through all the atoms and determine how many of them lie at each 0.5 increment

2. Use the total values at each increment to create pointers into an array

3. Fill up the array

For Example:

Suppose we have 7 atoms with the following coordinates (we will consider the

viewer to be looking down the z-axis and at a highly negative z value. Thus z = 30.0 is farther away from the viewer than z = 20.0)

ATOM No.	X	Y	Z
1	0.5	2.0	7.4
2	0.7	3.2	6.4
3	1.5	4.5	6.2
4	0.7	8.5	4.1
5	0.3	1.0	7.5
6	0.9	2.1	4.6
7	1.4	6.2	6.3

So, it can quickly be determined (in one pass) that there are :

2 atoms in the range z=7.0-7.49

0 atoms in the range z=6.5-6.99

3 atoms in the range z=6.0-6.49

0 atoms in the range z=5.5-5.99

0 atoms in the range z=5.0-5.49

1 atom in the range z=4.5-4.99

1 atom in the range z=4.0-4.49

And so, the "sorted" order of atoms from farthest to nearest would become:

1,5,2,3,7,6,4

Notice that this is not completely sorted, it is just sorted down to the 0.5 A level. This is good enough for a back to front drawing of atoms, and is very fast (takes approximately 2 passes through the data).

Some graphics cards may not require any sorting at all, as they implement what is known as a "z-buffer". Basically each pixel on the screen is initialized to a large "far away" value, and then as graphics are drawn to the screen the effective z's of the pixels being drawn are calculated and compared to the current z value of the pixel on screen. If the pixel being drawn is farther away than the pixel already on screen then nothing is done, otherwise the new pixel is drawn and the z buffer is updated with the new value. This commensurately requires more effort than simply blasting a two dimensional bitmap on to the screen (known as "blitting"), and so in some cases sorting then blitting is faster than a z buffer. However, on some graphics cards this z buffer routine is extremely fast, and clearly if a z buffer is used then no sorting is required, and thus could be faster in those cases.

MD simulations typically involve thousands or more atoms and thus particular attention should be paid to possible optimizations such as the sort above when creating a

visualization application. In the near future almost all graphics operations will be hardware based, and so the "fine art" of optimizations like sorting is likely to become outdated. But for the time being, knowing how the hardware works and when implementing a fast software approach can help is advantageous to the MD researcher and indeed anyone desiring fast graphics.

6.3. Animation of MD Simulation Data

Sets of static pictures can often reveal behavior of an MD simulation, but some phenomena only reveal themselves during animation and are difficult to capture in a series of slides. Animating MD data requires a speedy approach to graphics, as typically many thousands of atoms are being drawn per frame. This section discusses general approaches to fast animation that can be applied to MD simulations.

The first thing that needs to be done is acquire the MD data. Typically this data is contained within a series of frames taken at different instants of the simulation time. One could simply read in all x, y, and z positions of each atom for each frame. Then during animation the position of each atom could be advanced towards its next position by interpolating between two individual frames a certain amount. However, a more efficient approach would be to store the initial positions of each atom, and then store velocities of each atom for each subsequent frame (amount moved per interpolation between the

current two frames). This saves calculation time and would allow for better performance. Better still, if the animation is assured to be two dimensional in nature, one could convert the atom coordinates and velocities to screen space before storing them, thus skipping any necessary transformations during animation. Of course, if the "zoom factor" was desired to be changed during animation, then the values would all have to be adjusted again, but for static zoom levels this procedure would save even more time.

During any animation the atoms would have to be dynamically sorted, and so the sort from Section 6.2 would be a good choice (unless the graphics hardware possessed a faster implementation). This sort would have to be performed continuously during the animation and so minimizing the calculations is important. For 20,000 atoms, the sort described in Section 6.2 requires about 40,000 steps. If a frame rate of 24 fps (frames per second, which is the standard for motion pictures) was desired during the animation, then 40,000*24=960,000 steps are required per second just to sort the data. From this fact it can be seen why any possible optimization should be pursued in computer graphics, if good performance is desired.

The basic procedure for animation of MD simulations (in a 2D manner) can be summarized as follows:

1. Read in all the data. This can take up considerable space. For example, if there are

20,000 atoms distributed across 30 frames of data, then there is a total of 20,000*30*3=1,800,000 x, y, and z values to be stored. This does not include extra information, such as atom type, potential energy, etc. that might be desired to be known about each atom. All of this data has to be in the RAM for good performance, and so thought should be given at this stage as to how much data is required.

2. Perform transformations upon the data for speed. This includes replacing x, y, and z amounts (except for initial positions) with velocities. These velocities are basically linear interpolations between each frame. If nonlinear interpolations are desired, then more data needs to be stored (typical interpolations for trajectories are third order, i.e. splines). Nonlinear interpolations require more computations, and thus slow down the animation process. So, consideration of how far apart (in terms of simulation time) the frame data is (nonlinear interpolation does better for data that is far apart than linear interpolation) needs to be used in conjunction with the storage/performance penalties to decide which is a better approach.

3. Begin animation by drawing all the atoms to an offscreen surface and then displaying that surface while drawing the next one. One should never draw directly to the screen during animation, as this will be noticeable to the viewer. The technique of drawing offscreen and then displaying that screen when finished while drawing to the next screen is known as "double buffering" or "page flipping".

The approach described above was implemented to create a general purpose animation program. The program possessed the following features:

1. Designed for 2-D visualization. Since it was intended to view current MD simulations at high speed, a 2-D approach was taken. A 3-D viewer requires more computations (and thus slower performance, see 3-D discussion below) and was not necessary for the simulations performed in the present investigations

2. Allowed for zoom, change of atom size, color coding with various parameters such as depth, grain, etc. Also allowed pictures to be taken at various points and saved as bitmaps. See Figures 23 (a) - (d) for examples of these features.

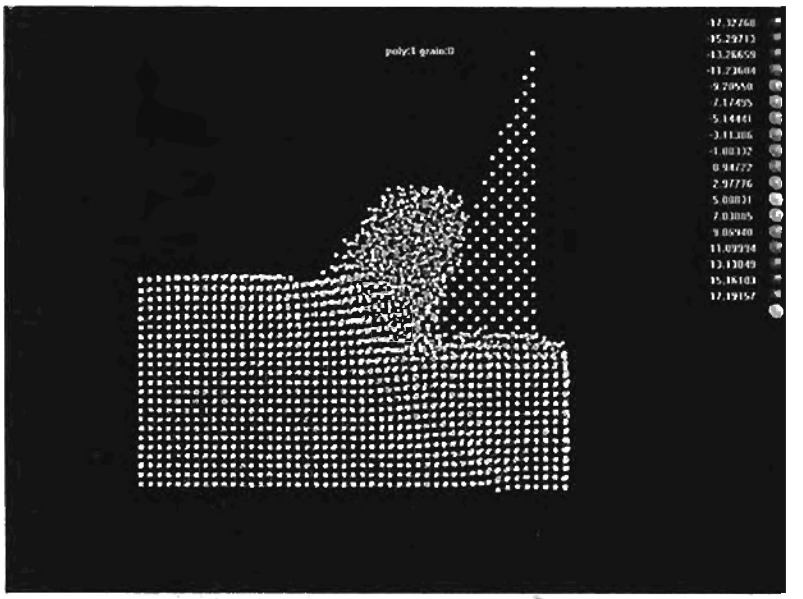


FIGURE 23a : Example of color coding with depth. This can help to reveal motion in successively deep planes into the page.

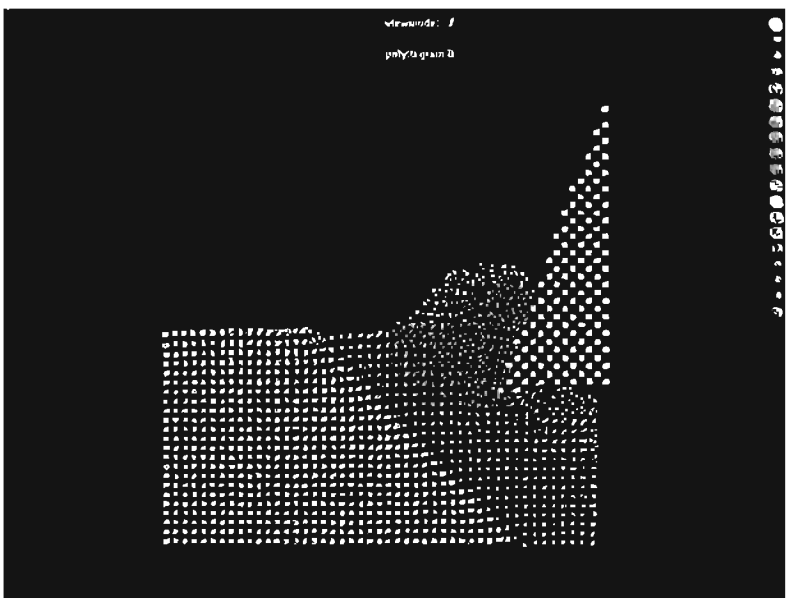


FIGURE 23b : Color coding by grain. This helps to delineate which atoms were part of a particular grain, especially after large deformations occur.

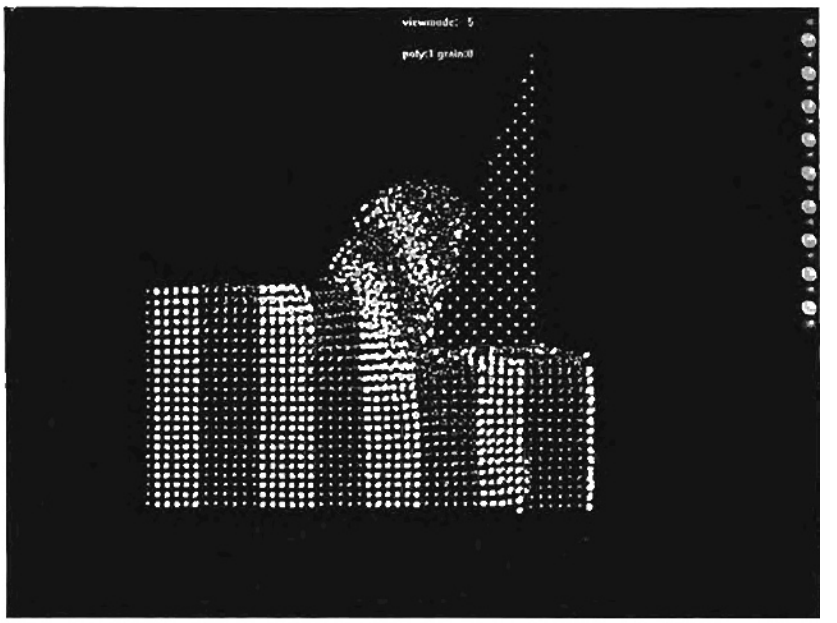


FIGURE 23c : Color coding by original horizontal position. In this way the behavior of individual layers can be monitored over time.

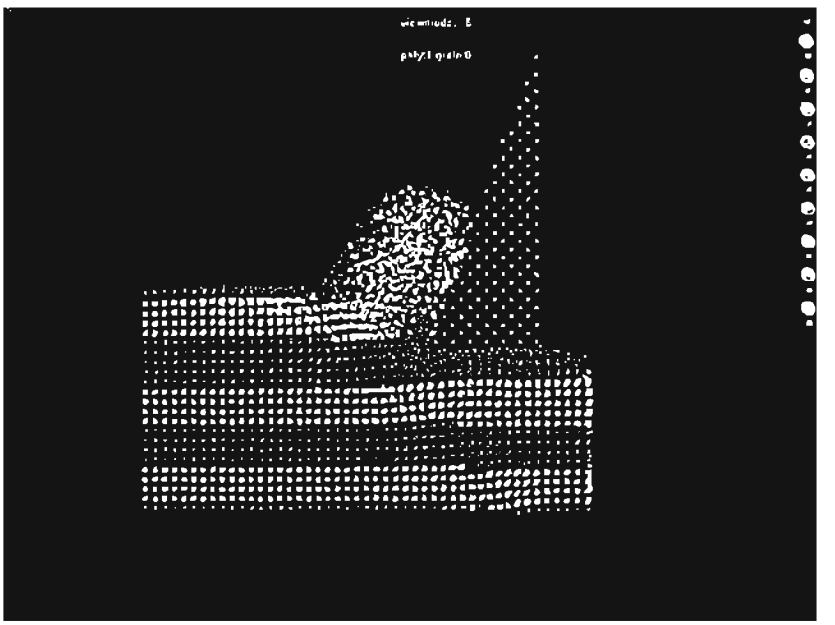


FIGURE 23d : Color coding by original vertical position. Once again, layers can be tracked during a simulation.

3. Was written under Microsoft's DirectX API for x86 Windows (PC) machines. This ensures maximum portability to any PC platform.

4. Overall code size was less than 300 kilobytes, and so easily fits on one floppy disk.

In the future, as the atom counts of MD experiments increase, a 3-D viewer might prove useful. 3-D applications require more mathematical transformations per atom, and so would be slower, Also, viewing large sets of atoms in 3-D it is difficult to see anything worthwhile in many cases (except on the surface) due to the fact that there are many atoms "in the way". In order for a 3-D application to be sufficiently useful, a very high quality atom (i.e. with dynamic light sourcing, etc.) needs to be drawn, and this requires still more horsepower. However, graphics hardware is beginning to support 3-D related operations directly, and so this is becoming more and more feasible on the average desktop PC.

6.4 Methods for Creating Atom Data in MD

In order to perform molecular dynamics simulations, atomic positions must be known beforehand. For simple shapes (such as cubes and pyramids for example) it is not

too difficult to devise an algorithm to "fill" such a shape with atoms. If more complex shapes are desired, then a more general algorithm is called for. The method developed below was used in the grain simulations to follow, and is effectively a 2-D/3-D hybrid way of generating complex shapes.

The general method used is one of clipping arbitrarily shaped polygons. This was selected due to the fact that a constant cross sectional area was deemed acceptable for the simulations to be conducted, and so a "true 3D" clipping was unnecessary. The basic principle of clipping an atom to a polygon is thus:

1. draw a horizontal line coincident with the atom

2. calculate how many times the line intersects and edge to the left of the atom

3). if the number of left intersection is odd, the atom is inside, else it is not

Figure 24 shows this procedure in action. Notice that the number of times that the horizontal line intersects the polygon to the left of the atom position is 3 (odd), and so therefore the atom lies within the polygon.

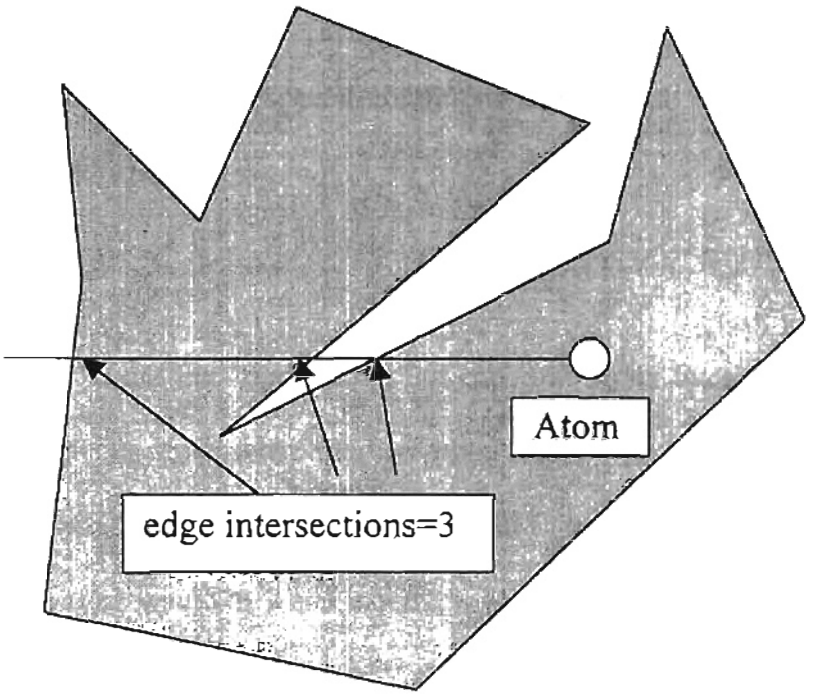


Figure 24 : Displays the concept of calculating edge intersections of a horizontal line to the left of the atom.

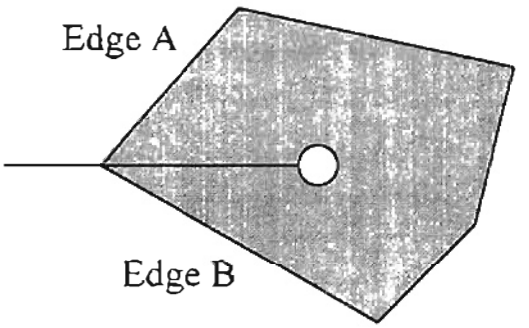


Figure 25 : Example of vertex intersection problem.

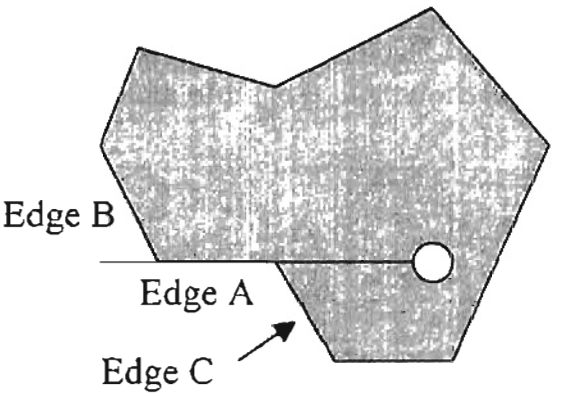


Figure 26 : Example of problem with horizontal edges.

There are a couple of special conditions that need to applied for this clipping procedure to work. They are:

1. Do not count the "minimum y" of each line as an intersection, as this would produce an erroneous value when the horizontal line crosses a vertex such as in Figure 25. If the horizontal line in Figure 25 was deemed to intersect both edges A and B, then the total for the atom would be 2 and incorrect. Therefore, since the intersection occurs at the minimum of A, only count B.

2. Do not count horizontal edges at all. Referring to Figure 26, if Edge A was included in the intersection calculations it would cause difficulty. Notice that Edge B is not included because the intersection is at the minimum of Edge B. Therefore Edge C is the only intersection, and so the total is 1 (odd).

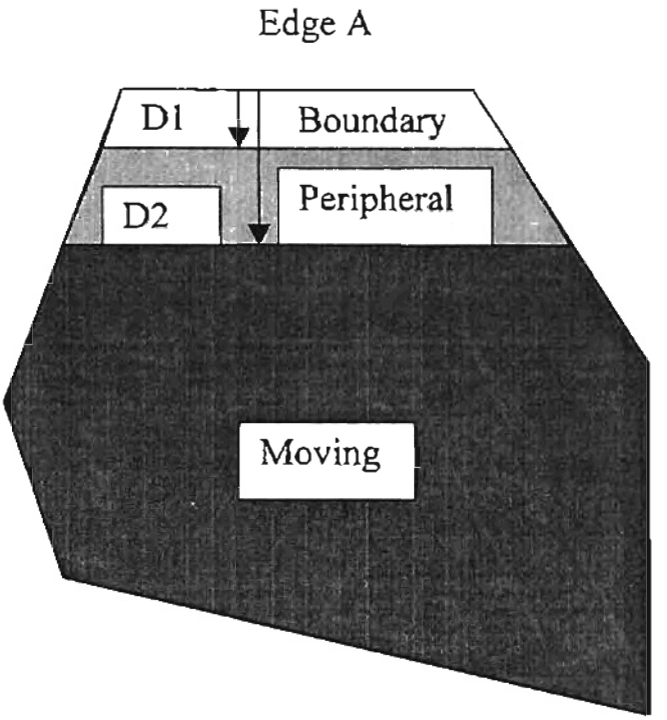


Figure 27a : Displays how the distance from an edge can be used to determine atom type.

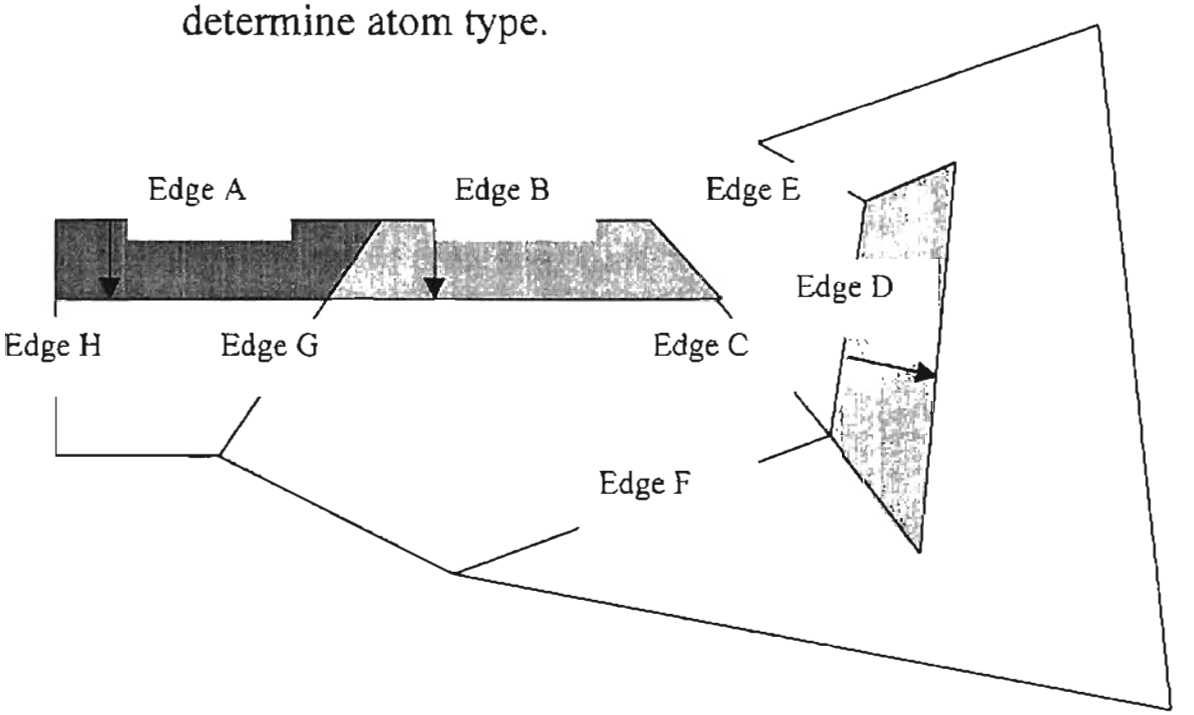


Figure 27b : Illustration of how to handle adjacent edges when calculating atom type from normal distance.

So, if one has defined a series of polygons defining crystals/grains and wishes to "fill them" with atoms, the above clipping algorithm works well. However, that is not the entire story. The earlier discussions of boundary and peripheral atoms (as well as "no-man's land", see Chapter 5) come into play next. The method to determine those characteristics used is simply defining normal distance from each edge wherein a particular type of atom exists. Figures 27 (a) - (b) display this approach. Figure 27 (a) displays the concept of determining an atom's type by its normal distance from a particular edge. Here any atom at a distance D1 or less from edge A is a boundary atom. It is peripheral if it is less than distance D2 and not less than D1. Figure 27 (b) shows how to determine which edge to use in the case of adjacent edges. If the adjacent edge to an edge is less than 180 degrees different, then the procedure is to use the adjacent edge as the "boundary" for the normal vector filling. Looking at Edge A, notice that Edges G and H are both less than 180 degrees, and so they are used as the boundary. Same case with Edge B (Edges C and G are less than 180 degrees). Notice how this allows the normal regions from A and B to mesh well. For Edge D however, both edges E and F are greater than 180 degrees. For this case, an average of the two adjacent edges is used for the boundary.

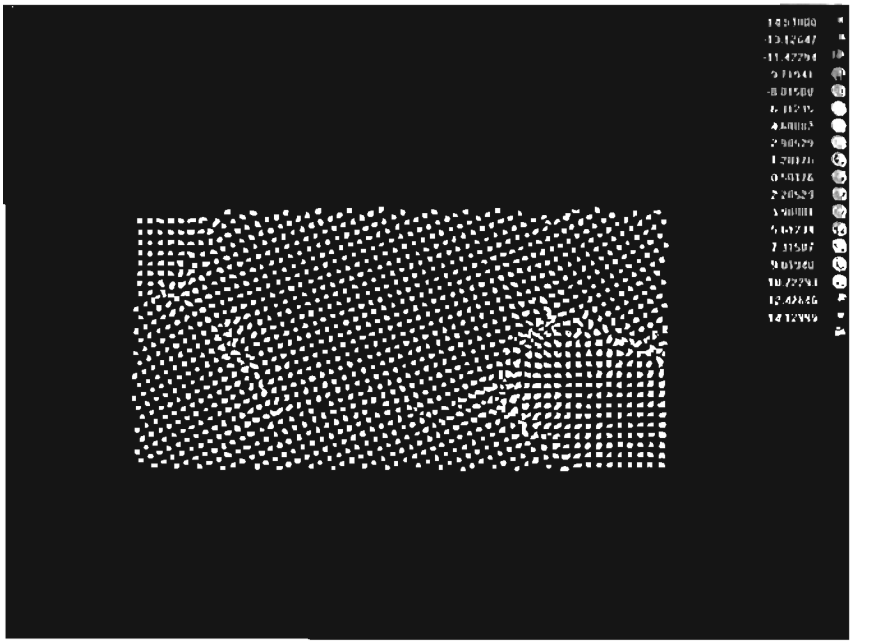


FIGURE 28a : Example of a multiple grained set of atoms made using the polygonal approach

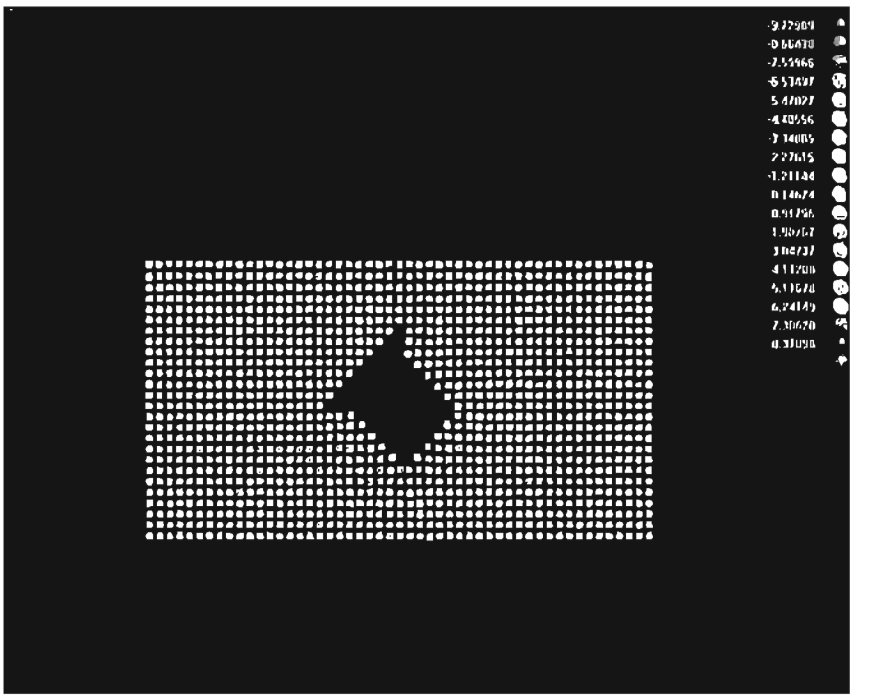


FIGURE 28b : Example of a set of atoms with a large initial polygonal shaped void in the center

Figures 28 (a) - (b) display some example MD simulations set up using the above described "polygon" approach.

6.5. Future Methods

The methods described above are adequate for small "2-D/3-D" simulations. For larger simulations where true 3-D shapes are required the algorithms will change somewhat but the approach will likely be similar. Usually the bottleneck in terms of design is the interface used. For constant cross section "2-D/3-D" simulations, a text based interface which inquires about the vertex positions of the polygons and which vertices make up which polygon is adequate. However, when moving to true 3-D, a graphical interface is demanded, as it is very difficult to properly visualize a set of three dimensional grains from merely seeing their vertices represented in x, y, and z numbers. Work is in progress at OSU on an "MD-CAD" program which will allow for a very quick and graphically based setup of MD simulations. It will include not only provisions for setting up atomic positions but also for directing the motions of the various bodies, selecting from a standard set of materials or customizing one's own, etc. If properly designed, this "visual" approach should greatly reduce snags at startup due to user error and thus increase the amount of MD simulations (and perhaps more importantly, "correct" MD simulations) performed.

CHAPTER 7

INTRODUCTION OF GRAIN BOUNDARIES IN MD SIMULATION

7.1. Current Views

Nanocrystalline (sometimes known as "ultra-fine grained") materials have of late become the topic of much interest. While there is not a generally accepted theory of nanocrystalline behavior, it is clear that such materials do not follow the classical Hall-Petch relation for micrometer and larger grain sizes. The Hall-Petch relation is given by:

$$\sigma_t = k_1 + k_2 d^{-0.5} \quad H_v = H_{v0} + k_h d^{-0.5}$$

where σ_t = yield strength, d = average grain size, H_v = microhardness and k_1, k_2 are constants. Typically the value of k_2 is positive, and so yield strength is predicted to rise with decrease in grain size. However, this is not the case when grain sizes approach the nanocrystalline level. Hall-Petch behavior fails to describe strength characteristics of nanocrystalline materials however (Zaichenko and Glezer, 1997). Quite often a negative value for k_2 is found at these grain sizes, and moreover sometimes no equation fits the

behavior. In some cases, there appears to be "critical grain size" wherein k_2 effectively switches sign, signifying a decrease in yield strength with further decrease in grain size.

Perhaps the variance in behavior of nanocrystalline materials is due to the varied ways in which they are produced and the fact that the grain boundary itself (as opposed to just the grain size) is now a factor when machining at this level. The grain boundaries of materials often are characterized as a separate phase known as the "grain boundary phase". The shear modulus of the grain boundaries can differ by more than an order of magnitude from the grain itself. The thickness of the "grain boundary phase" has been studied for nanocrystalline materials using electrical resistance measurements, as grain boundaries affect electrical resistivity. A Cu sample (avg grain size of 100nm) was found to have an average grain boundary width of about 2.1 nm (Islamgaliev, et al., 1997). The relationship between grain boundary width and the strength of the phase therein to the grain shape itself seems to be a possible reason for different material behavior (Zaichenko and Glezer, 1997).

Other attempts have been made to explain the deviation from the Hall-Petch relationship by the concept of a DFZ (Dislocation Free Zone) and crack behavior (Saito et al., 1990). Basically the fact that any crack generated in a nanograin is similar in size to the grain vs. cracks in large grains are generally negligible, and that there is a DFZ between the crack and the dislocation slip band is cited as the reason for the deviation from Hall-Petch behavior.

As mentioned above, there are several ways with which to produce nanocrystalline materials. Some of the most common methods are:

1) Severe mechanical deformation, usually in an alternating 90° orientation

2) Gas phase/sputtering condensation of nanoparticles then compression of those particles and CVD/thin film processes

In any event, each of the above processes is subjected to a vast array of variables. The act of compressing grains together will create a random array of high and low angle boundaries, while slowly "growing" them would likely produce generally lower angle boundaries. It is the wide difference in grain angle, grain shape, contamination resistance (nanocrystalline materials can be excellent "getters", and thus very susceptible to chemical contamination) and other factors which make such materials hard to generalize.

The uncertainties in actual structure make computational studies of nanocrystalline behavior desirable, as factors such as grain shape, grain boundary width, etc. can be varied and controlled. Once a basic understanding of the principles are gained, then they can be extrapolated to fit the more complex real world materials.

7.2. Computer Simulation Approach

The first thing to realize about creating a multiple grain system using MD is that the atoms created instantly pop into existence. They are allowed to reach a somewhat low energy state during a relaxation period, but this period is small in terms of real time (on the order of picoseconds), and so does not allow for the relaxation to occur for as long as it often does in the real world. Therefore, approaches need to be taken to try and get the atoms to start off as close as possible to a low energy state. It is at the grain boundaries where the differences in energies will manifest themselves most heavily. And so, dealing with atom placement with respect to the grain boundaries would seem to be the most important consideration.

Take for example two grains forming a low angle boundary. The boundaries and orientations of the grains are likely to be predetermined by the simulation requirements (i.e. the researcher wants to test the interactions of two grains at orientations O1/O2 and boundary shapes B1/B2). There are two methods proposed and investigated for grain boundary setup : offset and no man's land. They can be summarized as follows :

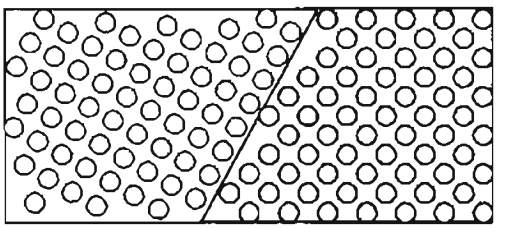


Figure 29 : Example of two grains.

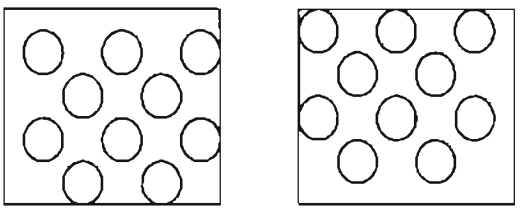


Figure 30 : Example of offset.

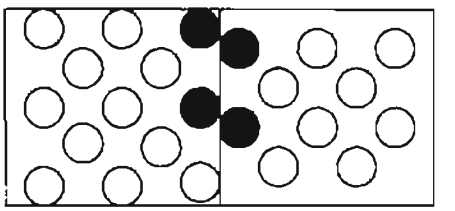


Figure 31 : Example of atoms that are too close together.

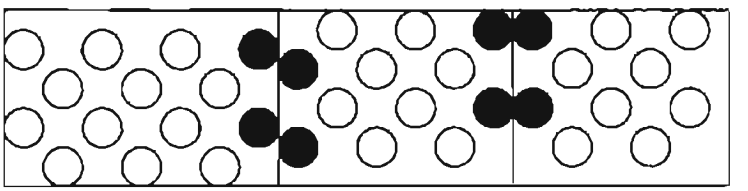


Figure 32 : Example of a situation where offset is unlikely to help.

Consider two grain boundaries as shown in Figure 29. The desired orientations of the grains within those boundaries are also shown. They are basically taken to be polygons with shared borders. Since the orientation of each grain was set from the beginning, the "offset" of each grain was used to get an initially low energy state. Offset is basically moving all atoms within a grain together an x, y, z amount, as shown in Figure 30. Notice that the atom positions have changed but not their orientations (angle). The motion was strictly a translation. The goal in using offset is to avoid a situation where two atoms in separate grains start too close together, as in Figure 31. The shaded atoms in Figure 31 are most likely too close together, and would possess a huge repulsive force between them. Offsetting either grain away from the other would alleviate this. Sometimes, however, the offset alone cannot accomplish this, as shown in Figure 32. In this case no amount of offset can help the middle grain. Offsetting the outer grains could help, but if they were "sandwiched" as well then offsetting them would not help. The "sandwiching" effect is more pronounced in 3 dimensions. Therefore, the concept of a "no-man's land" was implemented. This is a normal vector distance from an edge wherein no atoms are allowed to be placed. Figure 33 displays this concept in action. Both grains are not allowed to begin with atoms within a certain thickness of the grain boundary.

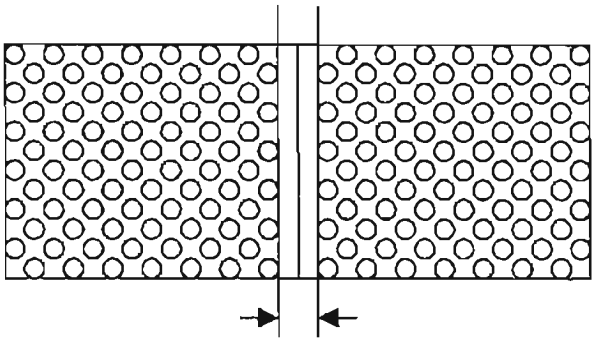


Figure 33 : Implementation of "no-man's land".

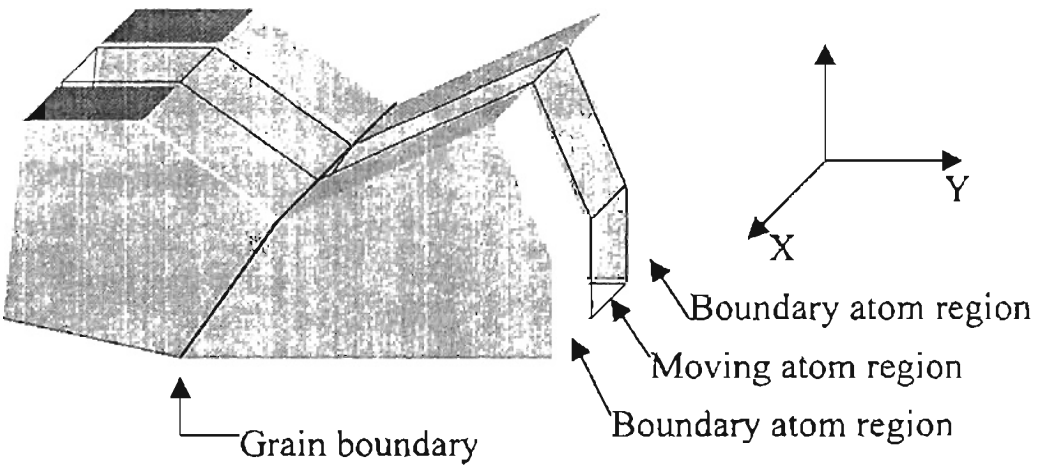


Figure 34 : Two grains with shaded boundary atom regions.

The important factor in guiding the next decision was simulation size. Recall from earlier that boundary atoms were implemented to stabilize the crystal structure in MD simulations. They were necessary because of the small number of atoms being considered (on the order of a few thousand). They were implemented again in this situation, on the positive and negative X faces of grains as well as any exposed outer edges that were desired to be fixed. The boundary atoms matched the orientation and spacing of the grain they were stabilizing and obeyed the same no-man's land restrictions as the non-boundary atoms (see Figure 34).

The selection of which configurations of atoms to test was influenced by a desire to see the basic influence of the following parameters upon deformation behavior in nanocrystalline materials:

1. grain size
2. grain boundary orientation
3. crystal orientation
4. relative grain-to-grain crystal orientation
5. no-man's land, and
6. depth of cut

In all cases the simulations were of similar size (about 15000 atoms) and generally took ~6 hours of CPU time to complete. The results of the simulations are detailed in the following section.

7. 3 Analysis of Results

The behavior of the various grain tests were characterized in terms of both visual deformation and force values. First the visual results are discussed and are broken up into categories based upon the specific tests being performed. A small diagram next to the text displays the structure which displayed those particular results.

Common to all simulations (unless otherwise noted) :

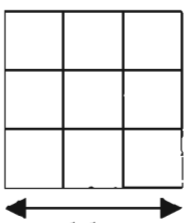
Tool: 20 degree rake, 5 degree clearance infinitely hard tool with neutral potential

Workpiece material: Cu (FCC: lattice constant 3.62 Å).

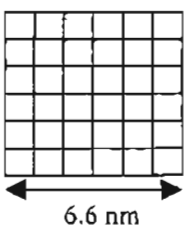
3 depths of cut tested: 1.1nm, 2.2nm, 3.3nm

"No man's land" equals 1 Å on each side of the grain boundary

Machined from right to left as pictured



1) *3x3 square grains all 0 degree orientation* (grain size=2.2 nm): Initial 45 degree stress lines after relaxation (see fig. 20). Displays 45 degree slip behavior as expected (45 degrees is major slip plane for FCC material oriented in a cubic fashion).



2) *6x6 square grains all 0 degree orientation* (grain size=1.1 nm): Very pronounced 45 degree slip behavior during machining, especially with the largest depth of cut (3.3 nm) Very well defined 45 degree shear zone ahead of tool tip as well. Atoms also exhibited "rolling" (quasicircular path of deformation of atoms near tool tip)

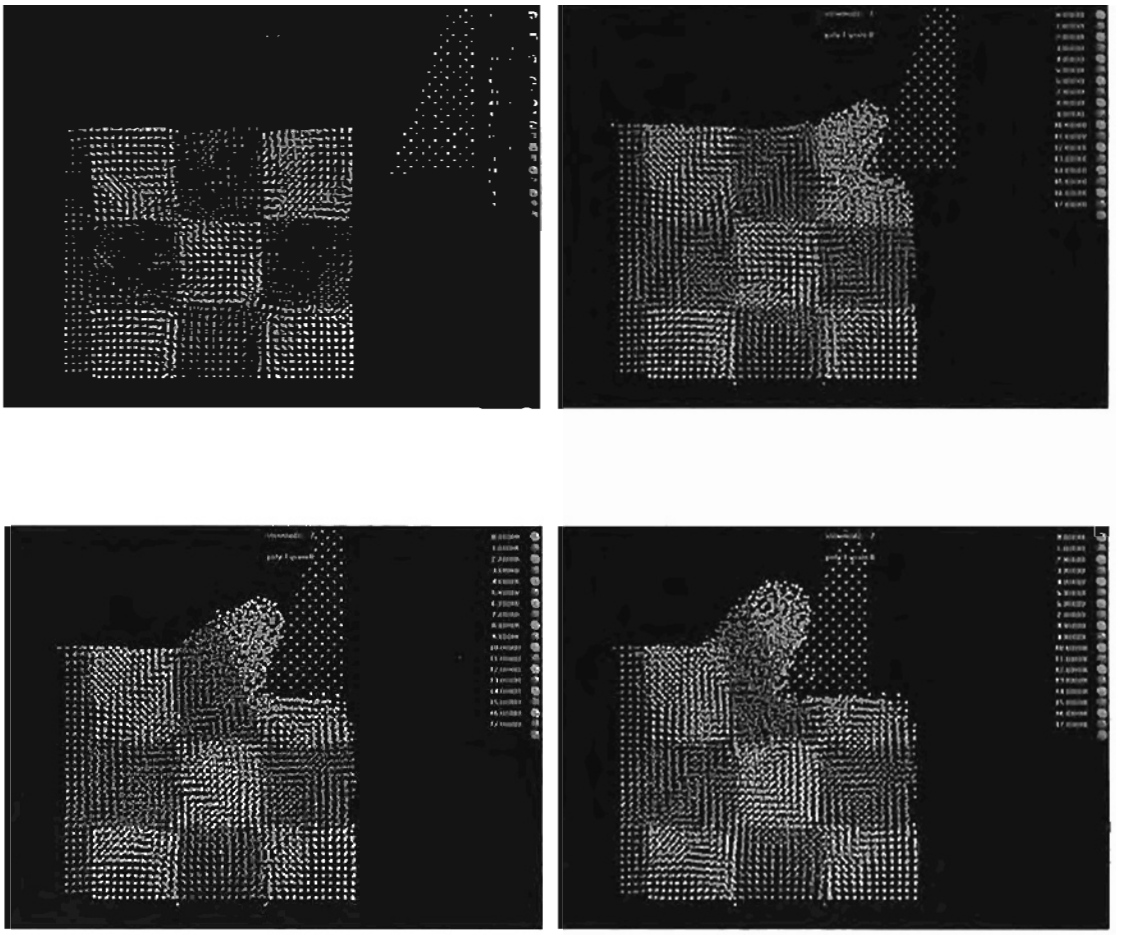


Figure 35a: Slides for Grain Test 1

MD simulation of nanometric cutting of Cu at various stages
All grains are cubic orientation
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

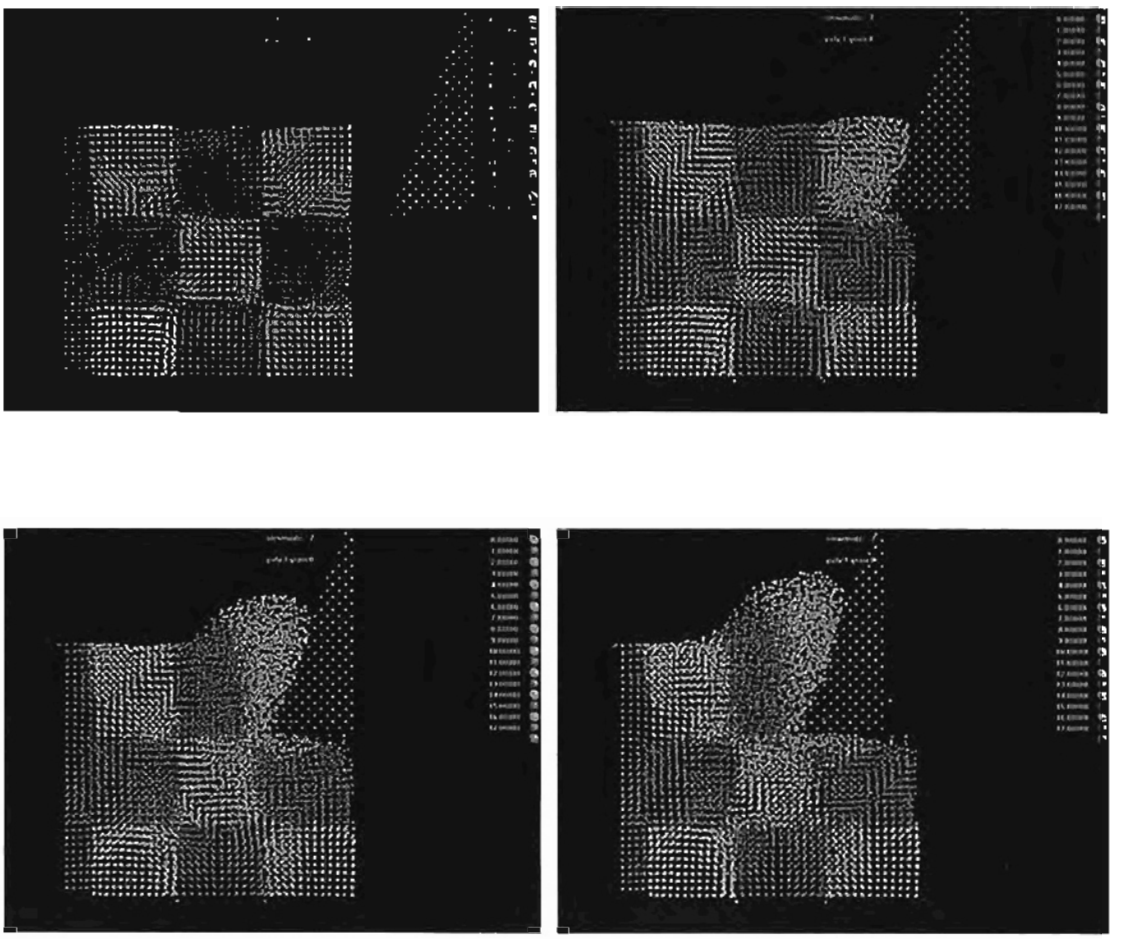


Figure 35b : Slides for Grain Test 1

MD simulation of nanometric cutting of Cu at various stages
All grains are cubic orientation
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

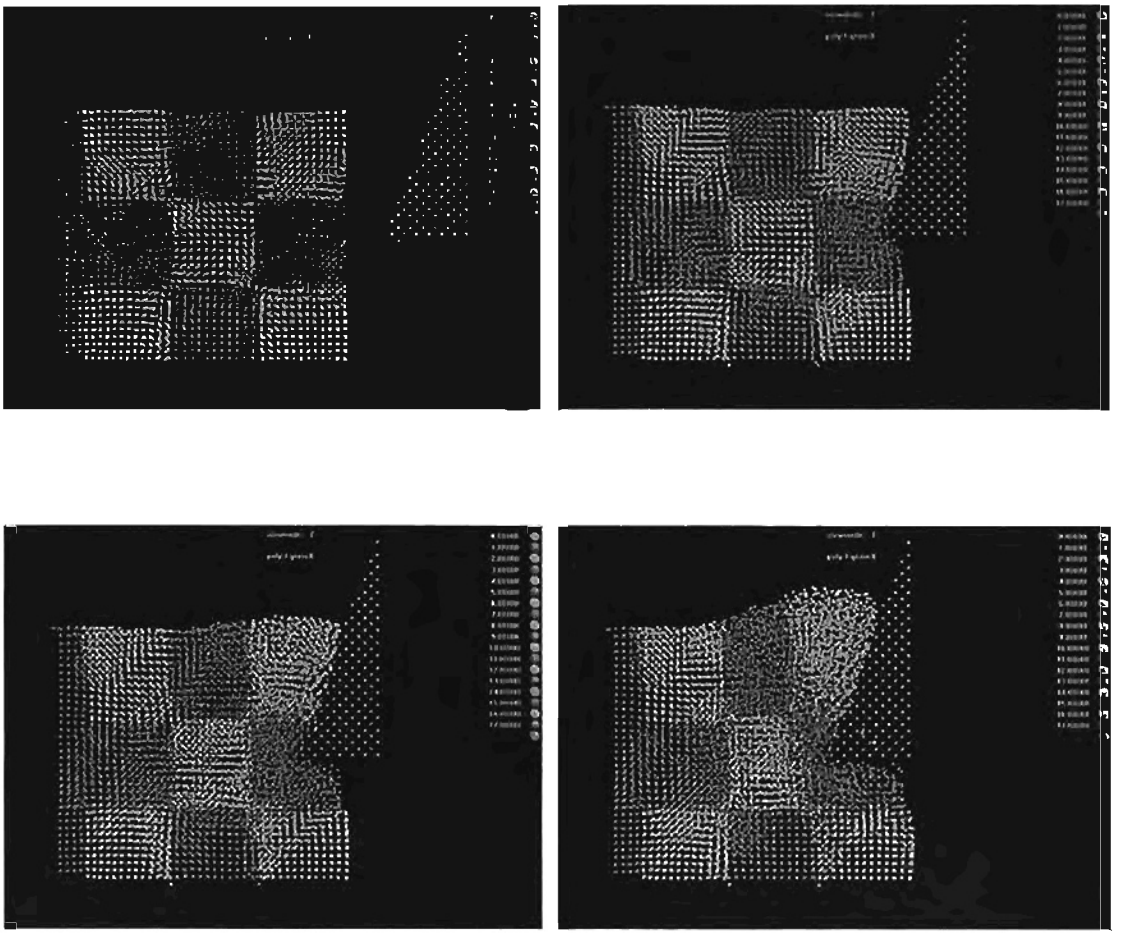


Figure 35c : Slides for Grain Test 1

MD simulation of nanometric cutting of Cu at various stages
All grains are cubic orientation
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

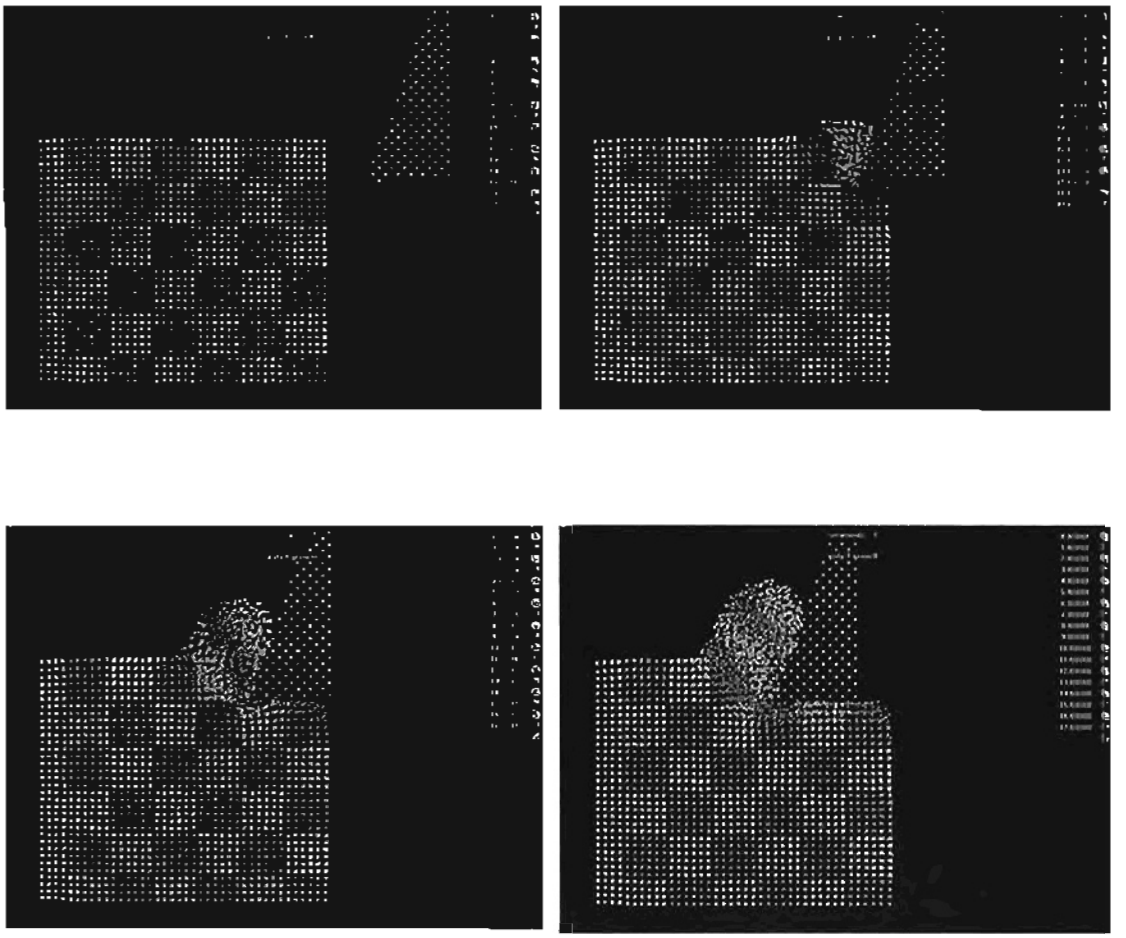


Figure 36a: Slides for Grain Test 2

MD simulation of nanometric cutting of Cu at various stages
All grains are cubic orientation
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

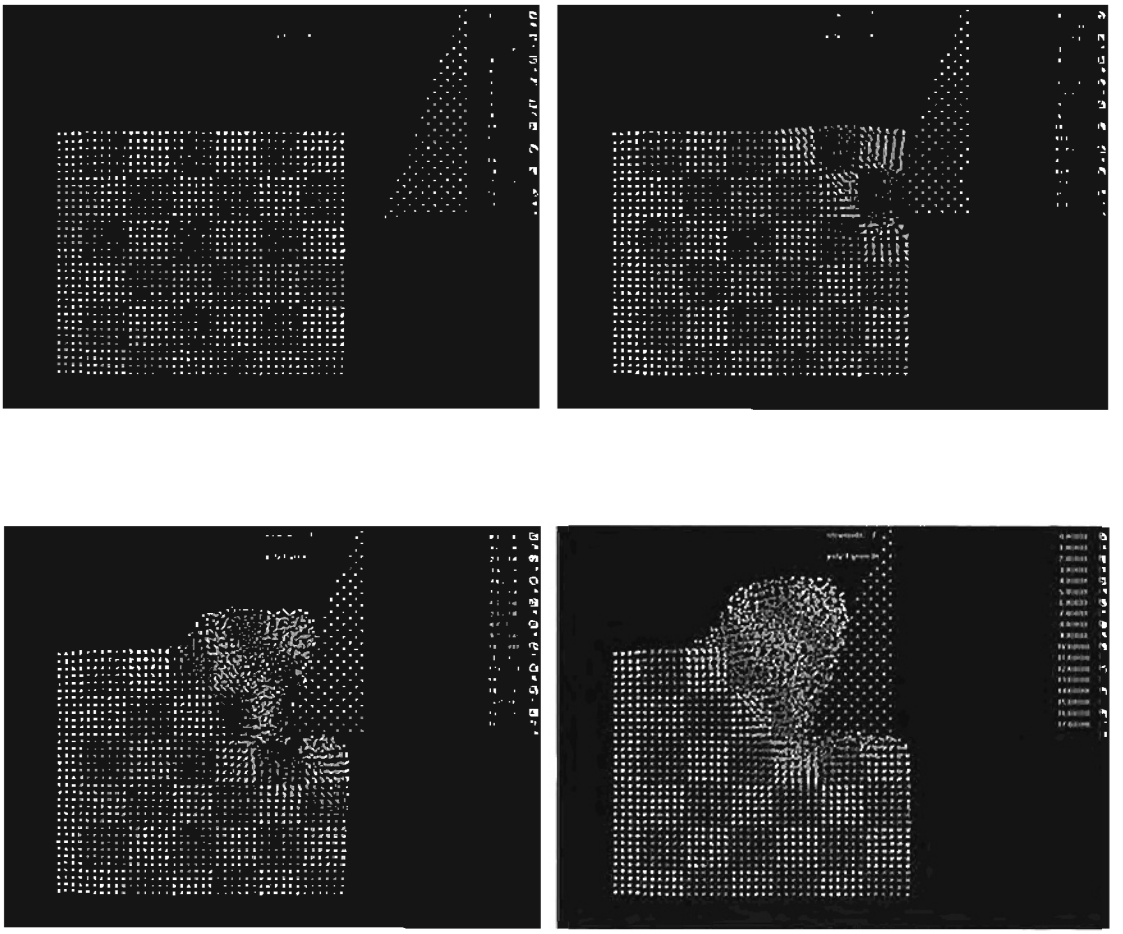


Figure 36b : Slides for Grain Test 2

MD simulation of nanometric cutting of Cu at various stages
All grains are cubic orientation
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

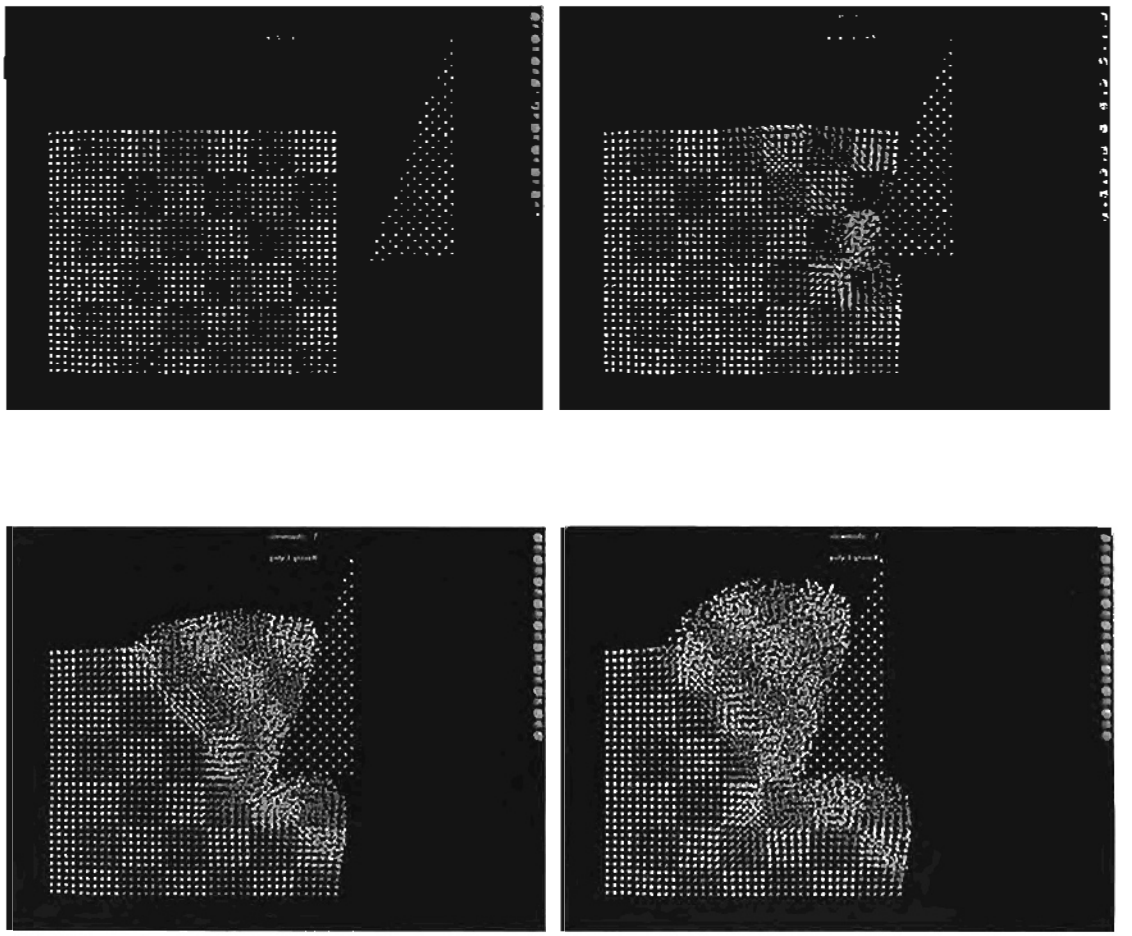
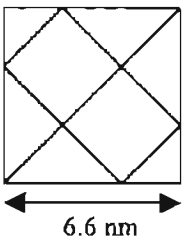
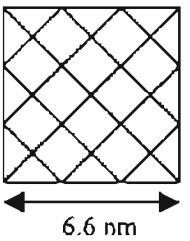


Figure 36c : Slides for Grain Test 2

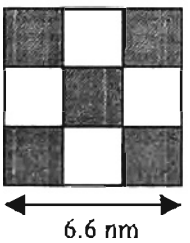
MD simulation of nanometric cutting of Cu at various stages
All grains are cubic orientation
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s



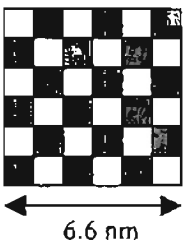
3) *Diamond shaped grains all 45 degree orientation* (grain size=3.1 nm): Displays 0/90 degree slip behavior (expected from the 45 degree orientation). "Rolling" begins early and stays constant for 2.2nm/3.3nm cuts, no real roll for 1.1 nm depth of cut. Roll is very pronounced for middle depth of cut (2.2 nm) at the first 4 corner junction.



4) *Diamond shaped grains all 45 degree orientation* (grain size=1.55 nm): Very pronounced 0/90 degree slip behavior. Rolling behavior starts early and stays except for shallow (1.1 nm) depth of cut, which takes longer to manifest itself.

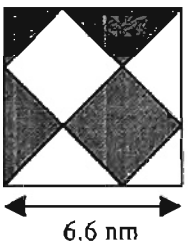


5) *Square grains with shaded grains 45 degree orientation, unshaded 0 degree:* 0 degree grains try to slip in 45 degree fashion, 45 try and slip in 0/90 degree. The tool/chip region distorts this behavior close to it. There is pronounced roll when crossing grain boundaries.



6) *Square grains with shaded grains 45 degree orientation, unshaded 0 degree*

No real definable slip behavior. Very constant rolling effect.



7) *Diamond grains with shaded grains 0 degree orientation, unshaded 45 degree:* Grains attempt to slip according to orientation of grain (0 degree grains in 45 degree fashion, 45 degree grains in 0/90 degree).

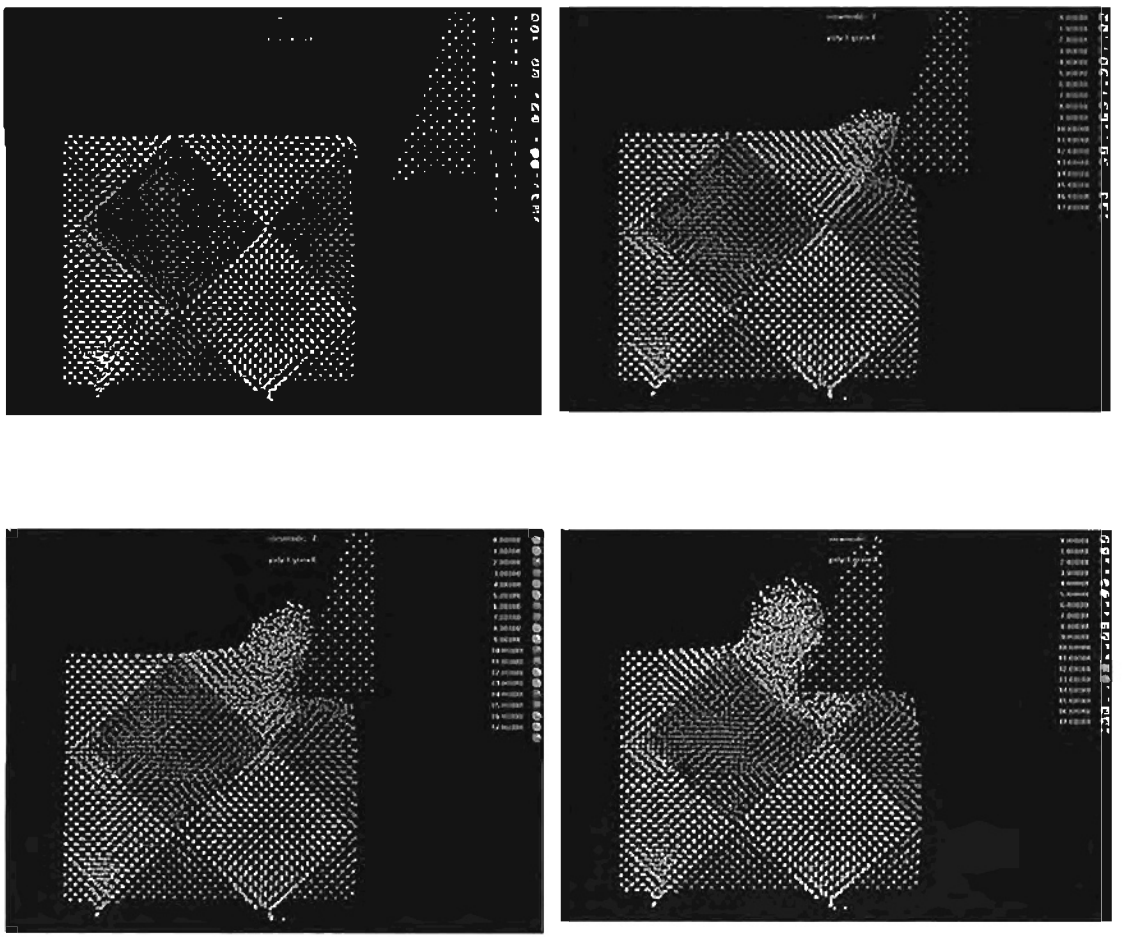


Figure 37a : Slides for Grain Test 3

MD simulation of nanometric cutting of Cu at various stages
All grains are 45 degree orientation
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

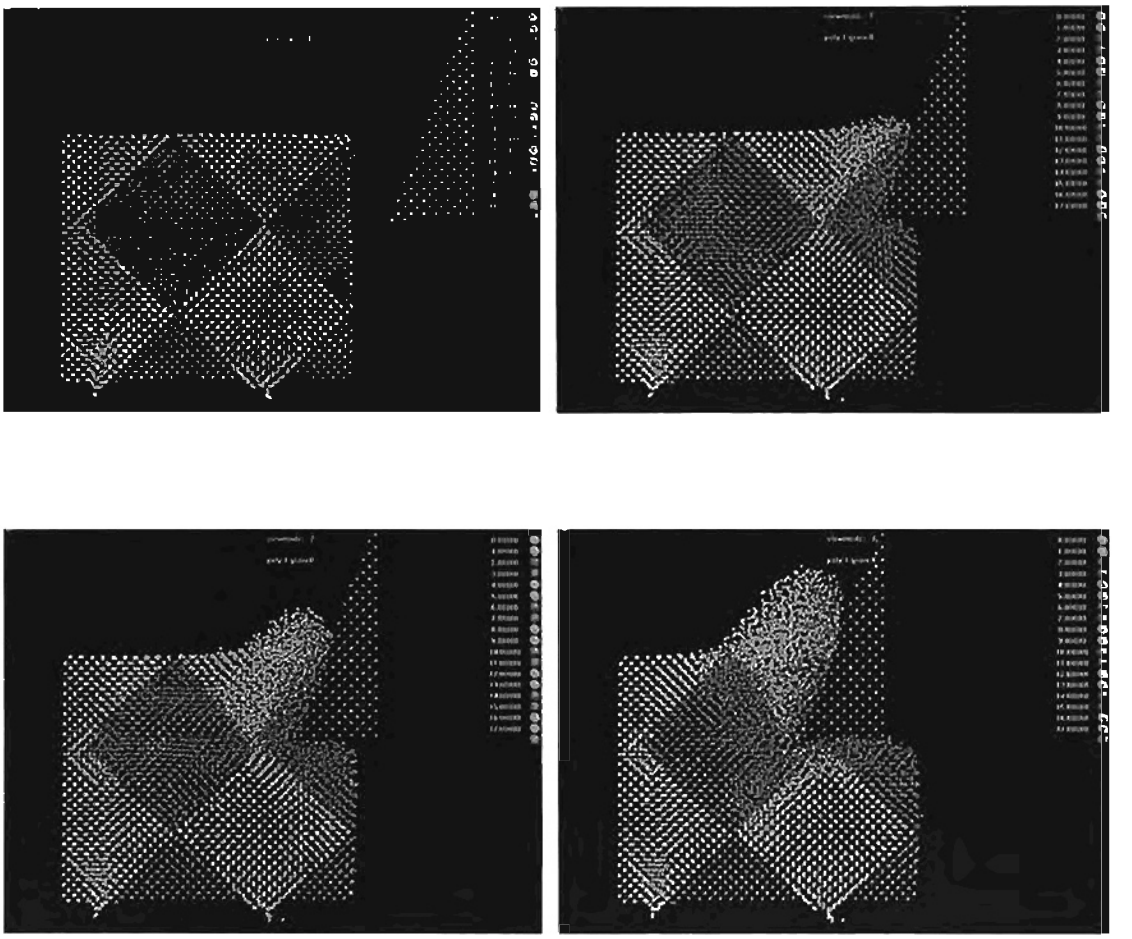


Figure 37b : Slides for Grain Test 3

MD simulation of nanometric cutting of Cu at various stages
All grains are 45 degree orientation
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

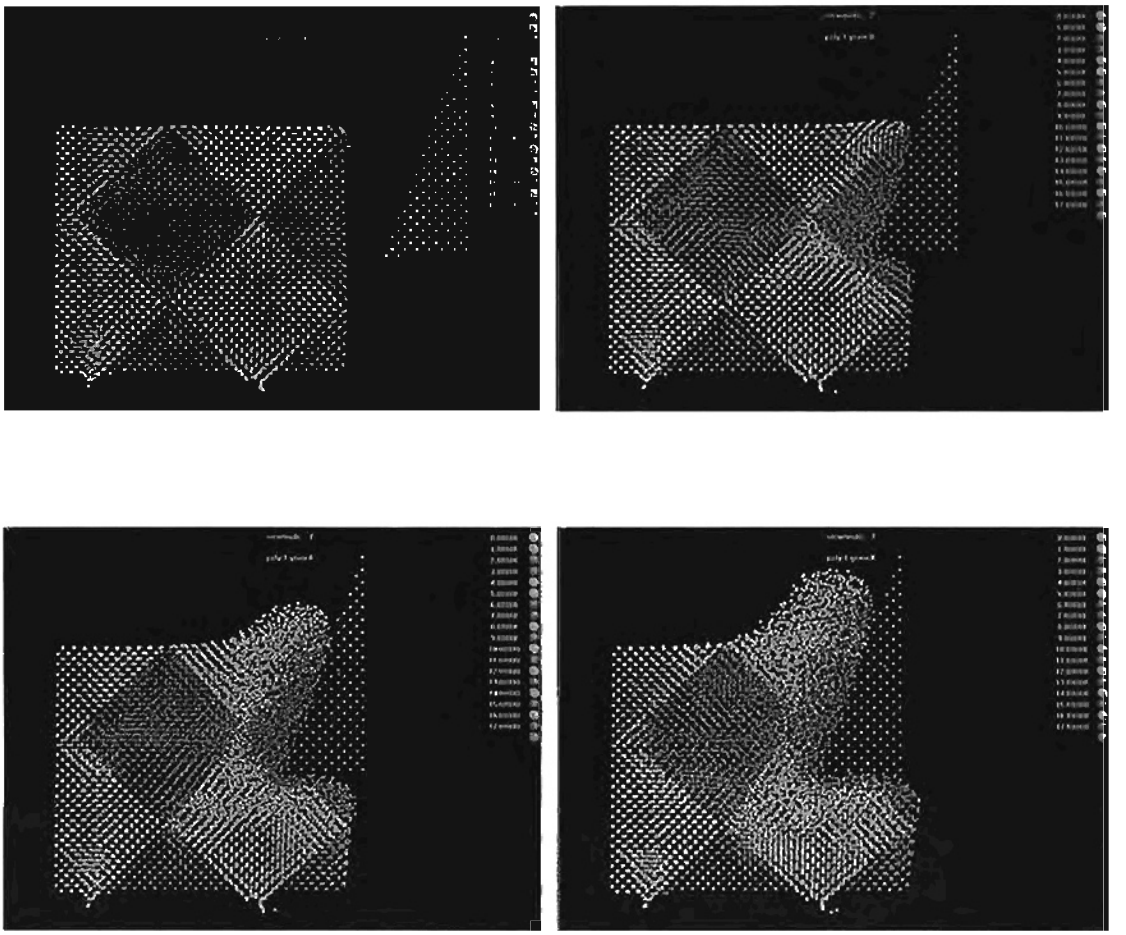


Figure 37c : Slides for Grain Test 3

MD simulation of nanometric cutting of Cu at various stages
All grains are 45 degree orientation
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

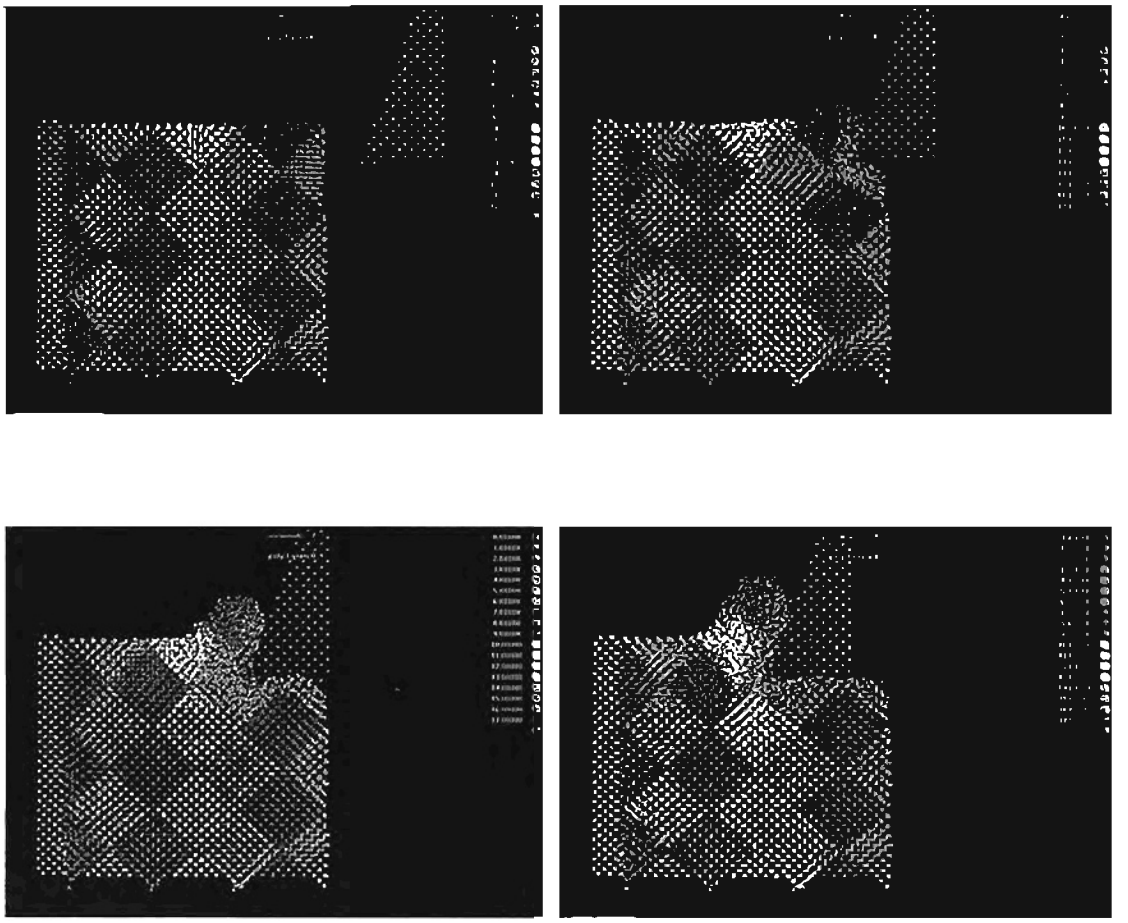


Figure 38a : Slides for Grain Test 4

MD simulation of nanometric cutting of Cu at various stages
All grains are 45 degree orientation
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

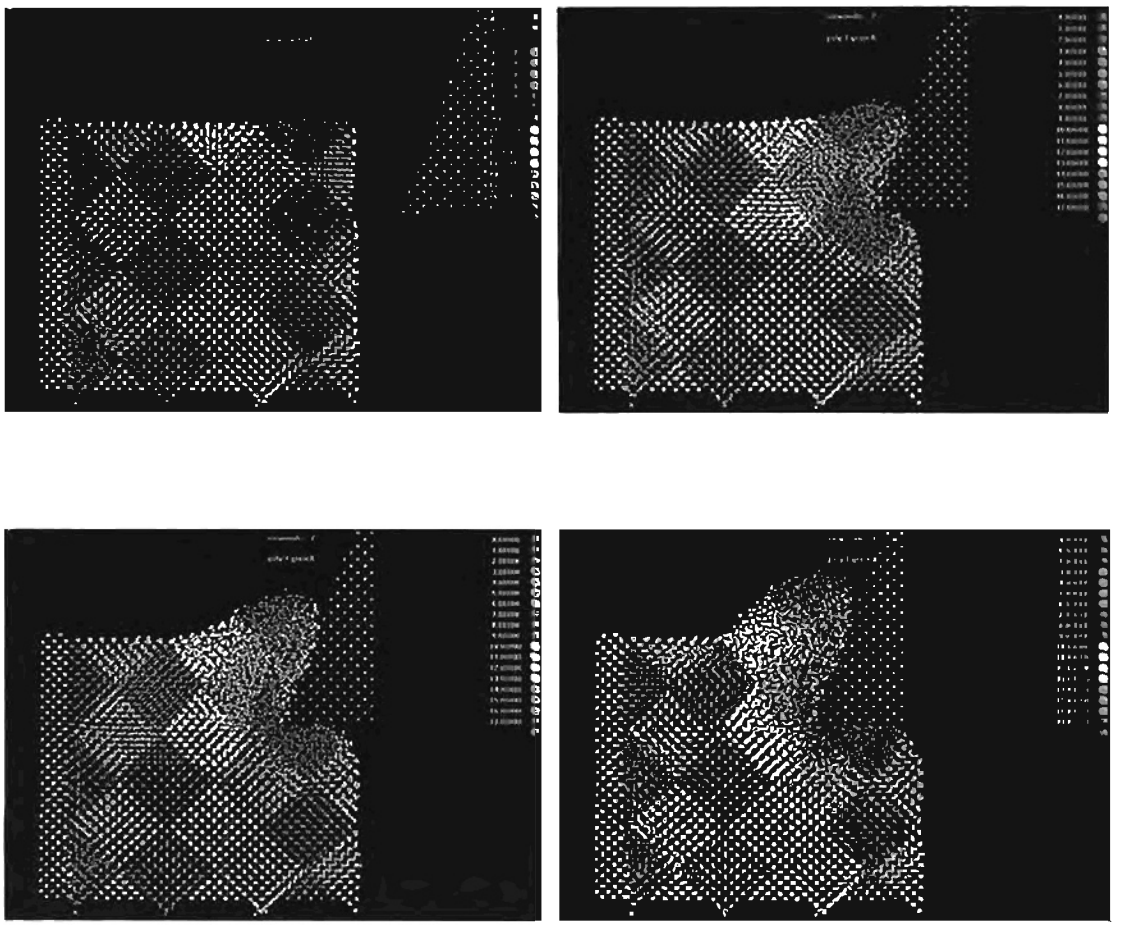


Figure 38b : Slides for Grain Test 4

MD simulation of nanometric cutting of Cu at various stages
All grains are 45 degree orientation
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

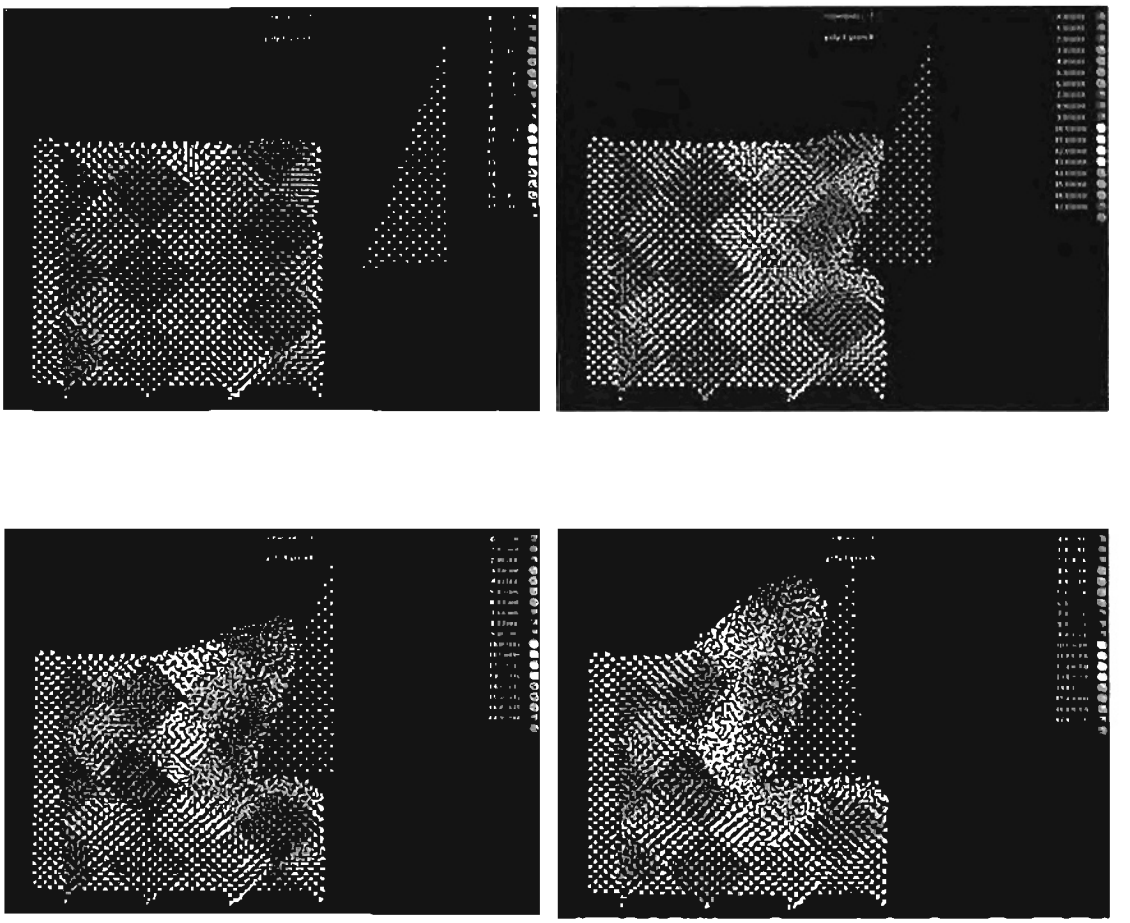


Figure 38c : Slides for Grain Test 4

MD simulation of nanometric cutting of Cu at various stages
All grains are 45 degree orientation
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

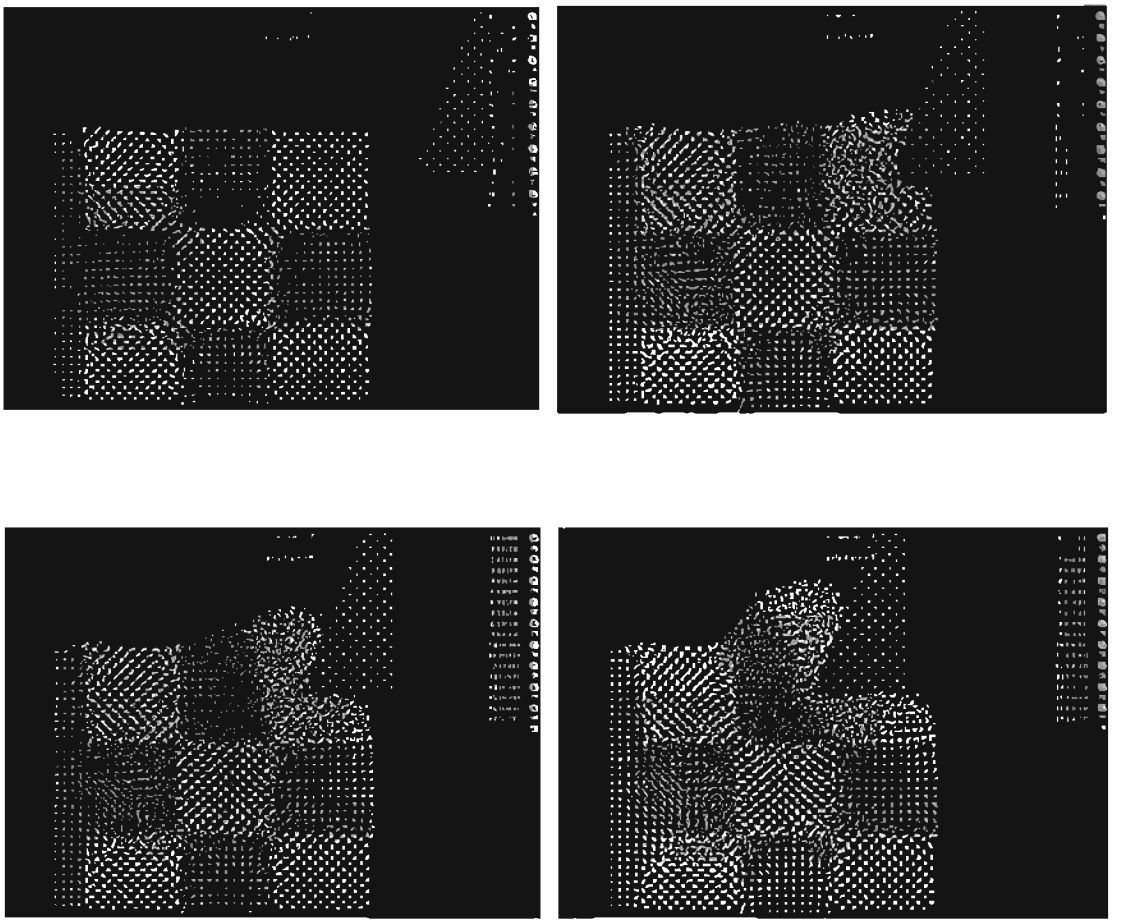


Figure 39a : Slides for Grain Test 5

MD simulation of nanometric cutting of Cu at various stages
Dark grains have cubic orientation, light are 45 degrees
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

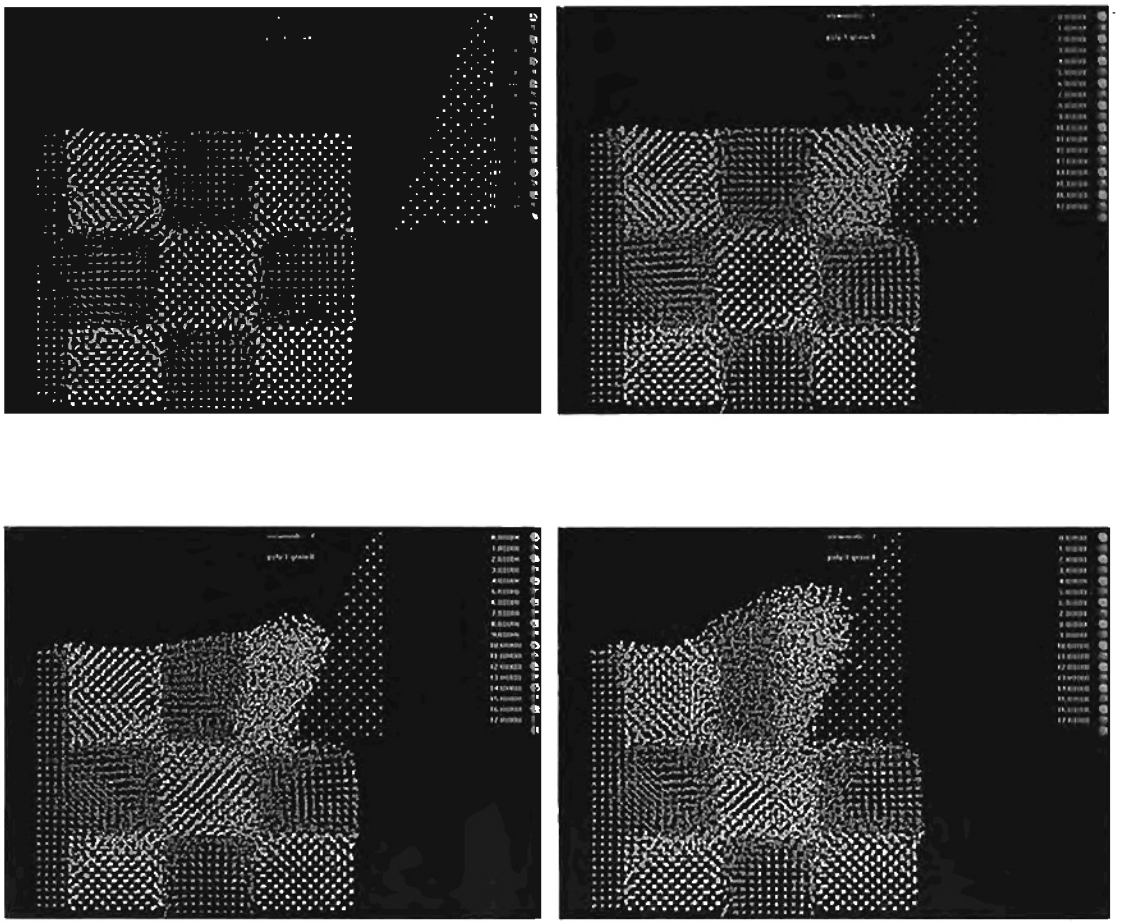


Figure 39b : Slides for Grain Test 5

MD simulation of nanometric cutting of Cu at various stages
Dark grains have cubic orientation, light are 45 degrees
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

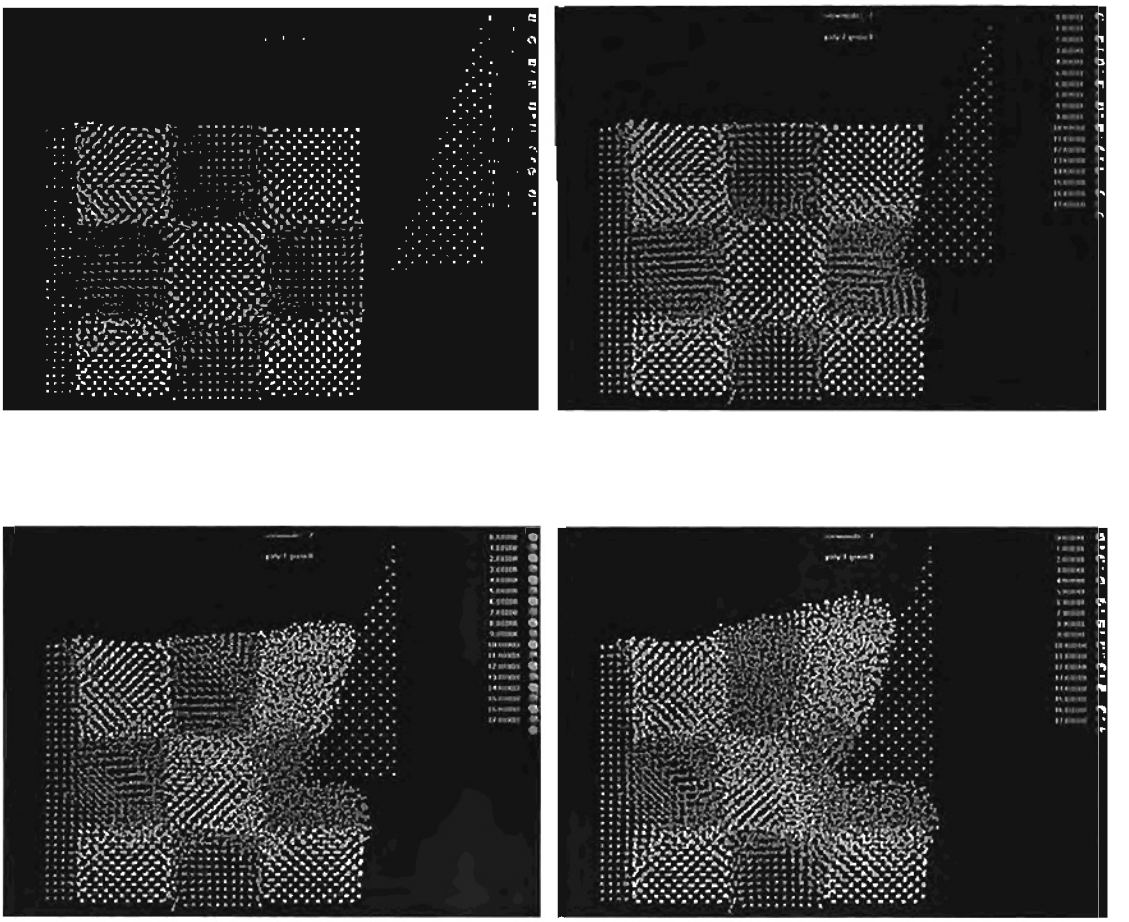


Figure 39c : Slides for Grain Test 5

MD simulation of nanometric cutting of Cu at various stages
Dark grains have cubic orientation, light are 45 degrees
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

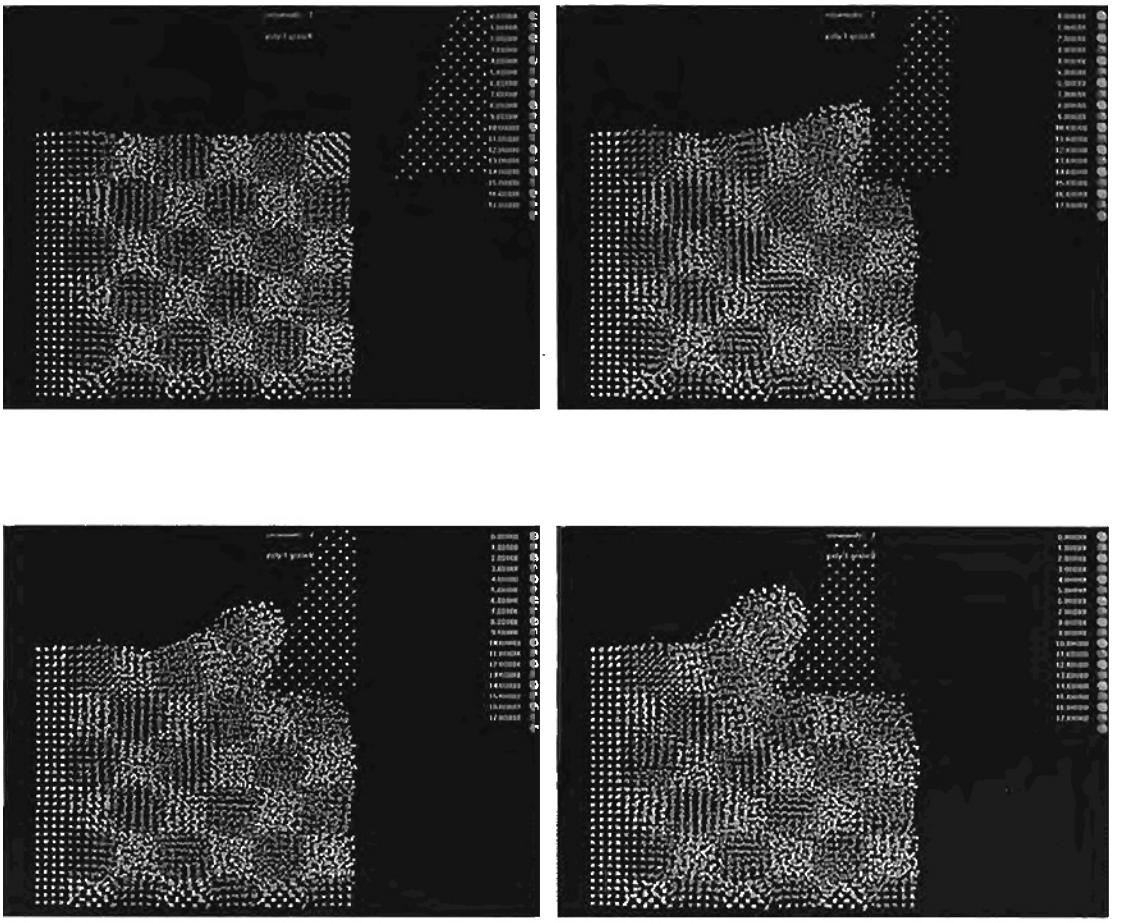


Figure 40a : Slides for Grain Test 6

MD simulation of nanometric cutting of Cu at various stages
Dark grains began at cubic orientation, light at 45 degrees
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

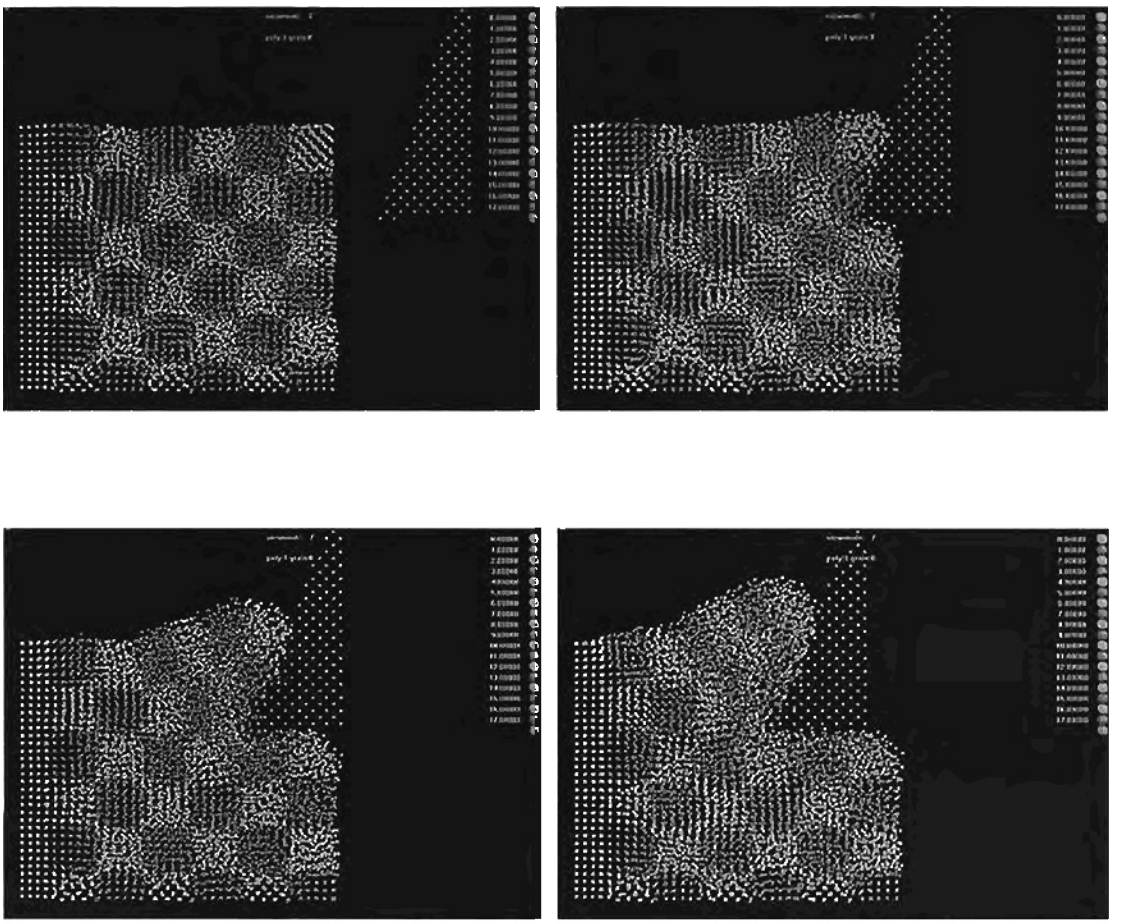


Figure 40b : Slides for Grain Test 6

MD simulation of nanometric cutting of Cu at various stages
Dark grains began at cubic orientation, light at 45 degrees
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

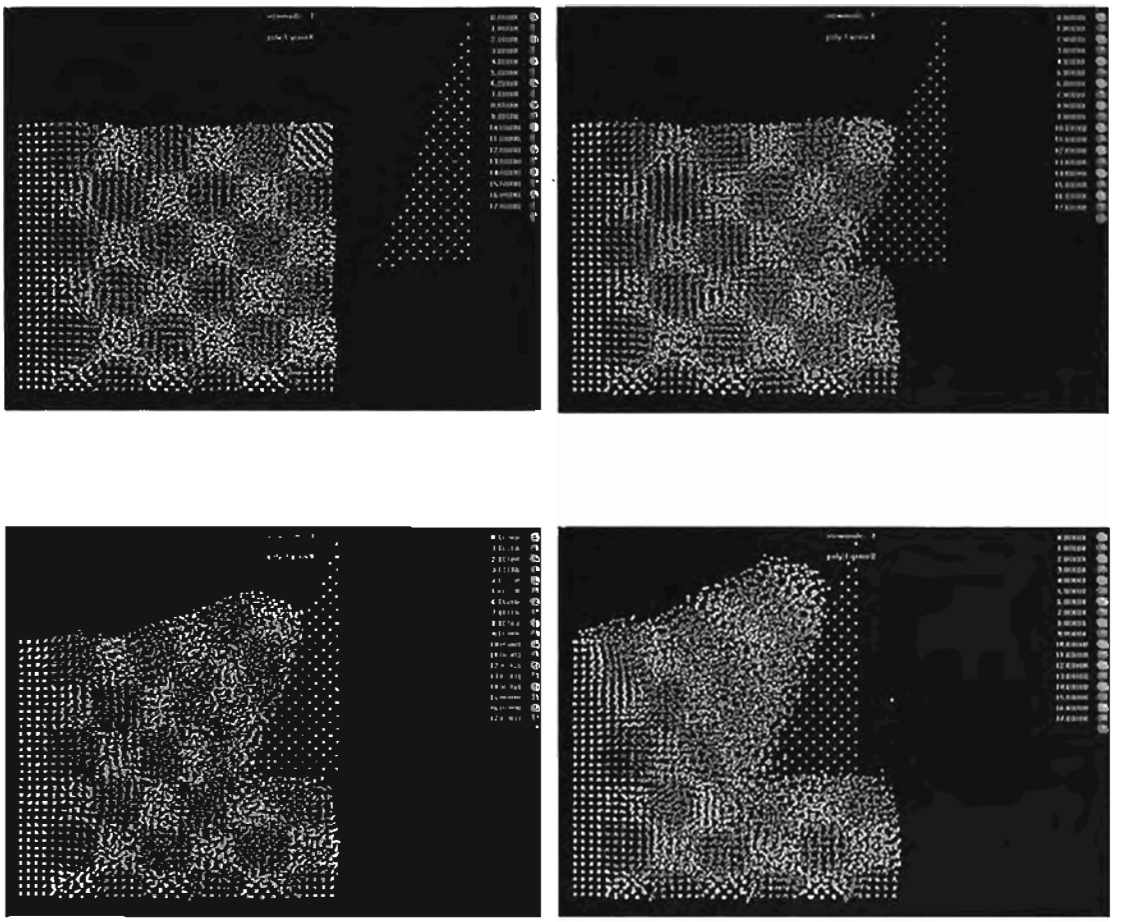
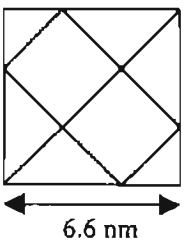


Figure 40c : Slides for Grain Test 6

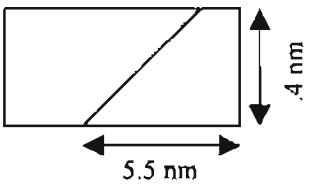
MD simulation of nanometric cutting of Cu at various stages
Dark grains began at cubic orientation, light at 45 degrees
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

Rolling behavior exhibited when crossing grain boundaries.



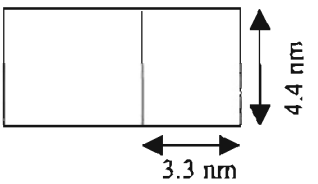
8) *Diamond grains all with 45 degree orientation and "no man's land" of 2.0 Å:* Displays 90 degree slip behavior. Very pronounced roll near boundary B for largest depth of cut (3.3 nm).

For the following simulations only one depth of cut was performed : 1.1 nm (with same 20 degree rake, 5 degree clearance tool)



9) *Two grains of 0 degree orientation sharing a 45 degree boundary*

Exhibits stress lines parallel to grain boundary after relaxation. Does not allow much slip parallel to grain boundary. Roll is present, especially across the boundary. 45 degree shear angle ahead of tool.



10) *Two grains of 0 degree orientation sharing a 90 degree boundary*

No significant slip behavior below tool. Roll is present, especially at grain boundary. 45 degree shear angle present.

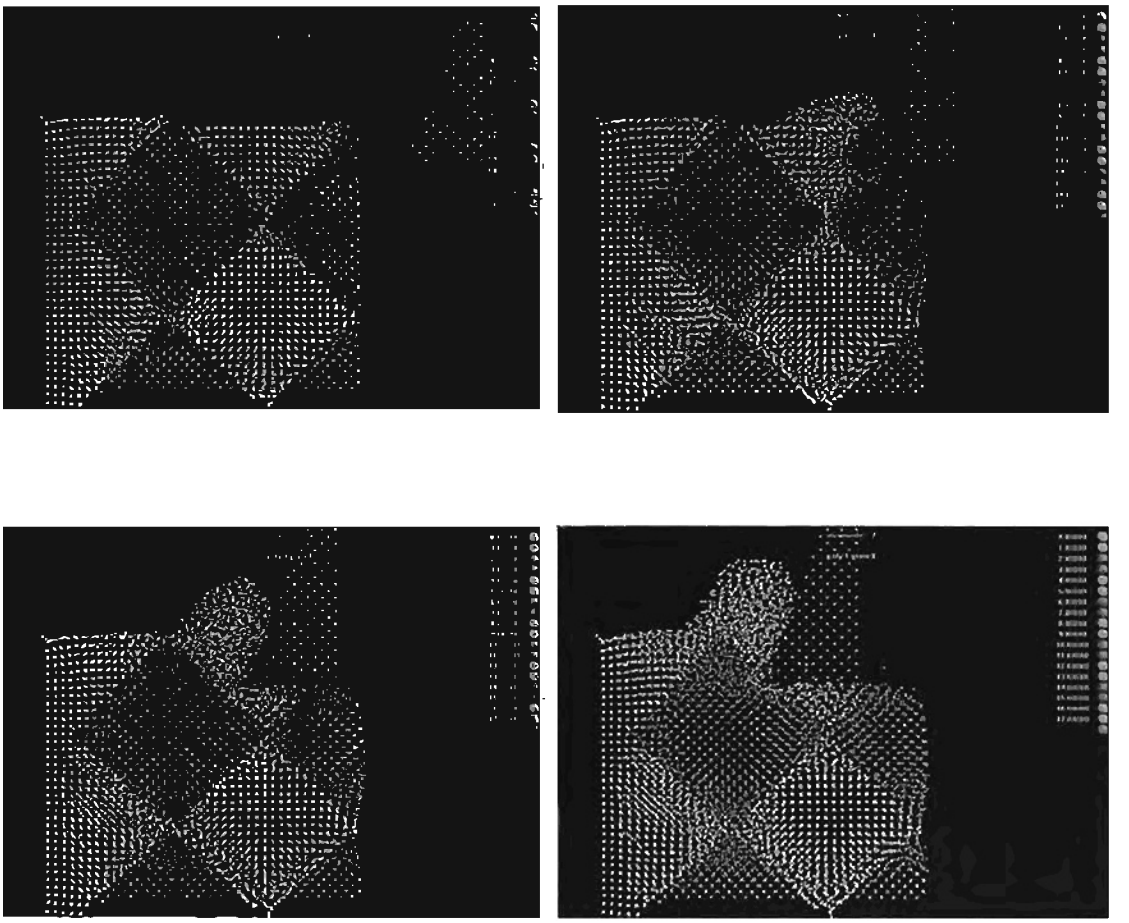


Figure 41a : Slides for Grain Test 7

MD simulation of nanometric cutting of Cu at various stages
Light grains have cubic orientation, dark are 45 degrees
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

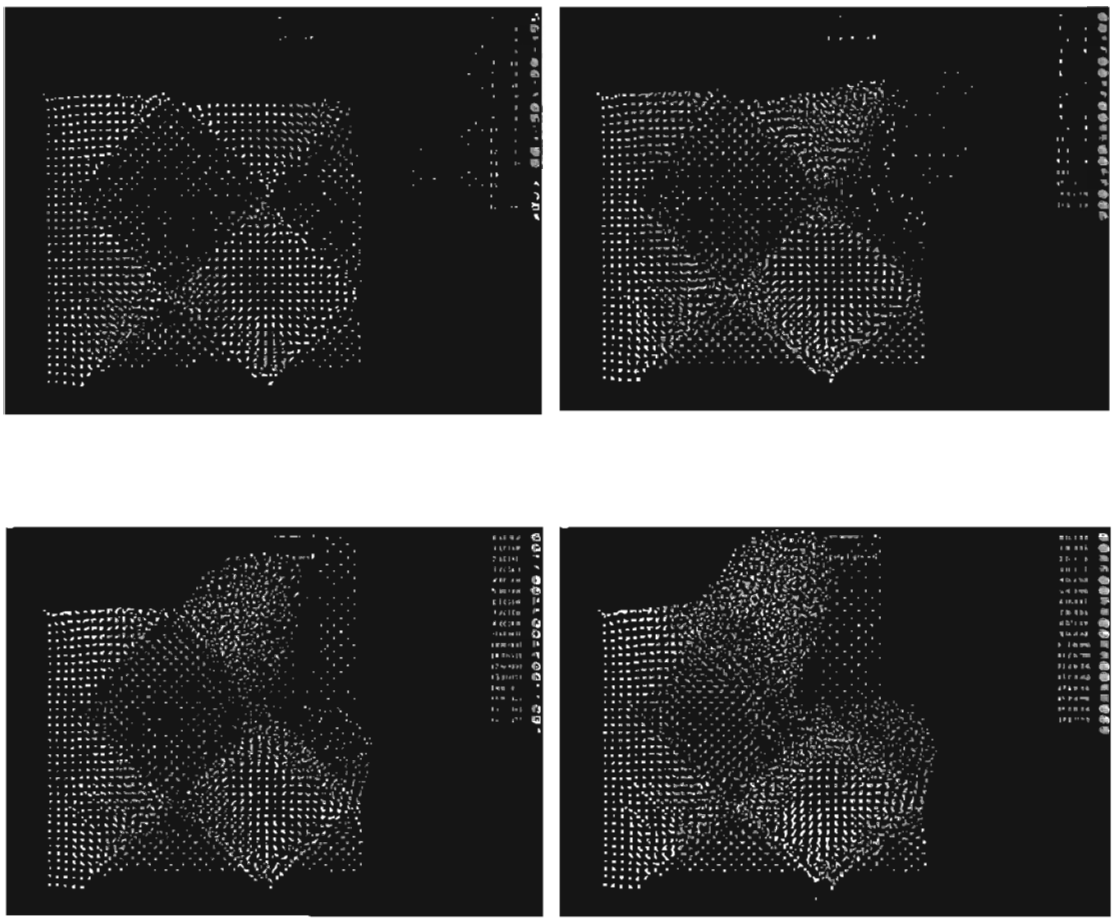


Figure 41b : Slides for Grain Test 7

MD simulation of nanometric cutting of Cu at various stages
Light grains have cubic orientation, dark are 45 degrees
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

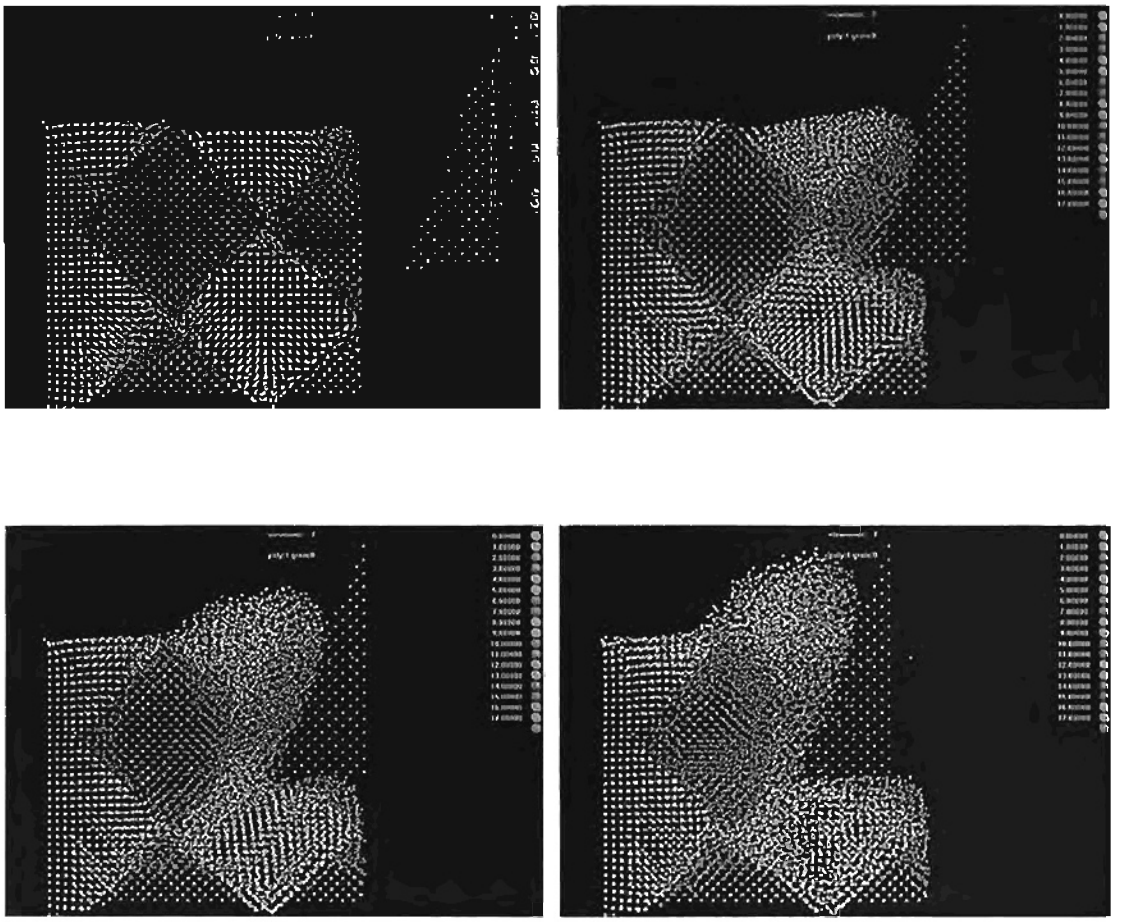


Figure 41c : Slides for Grain Test 7

MD simulation of nanometric cutting of Cu at various stages
Light grains have cubic orientation, dark are 45 degrees
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

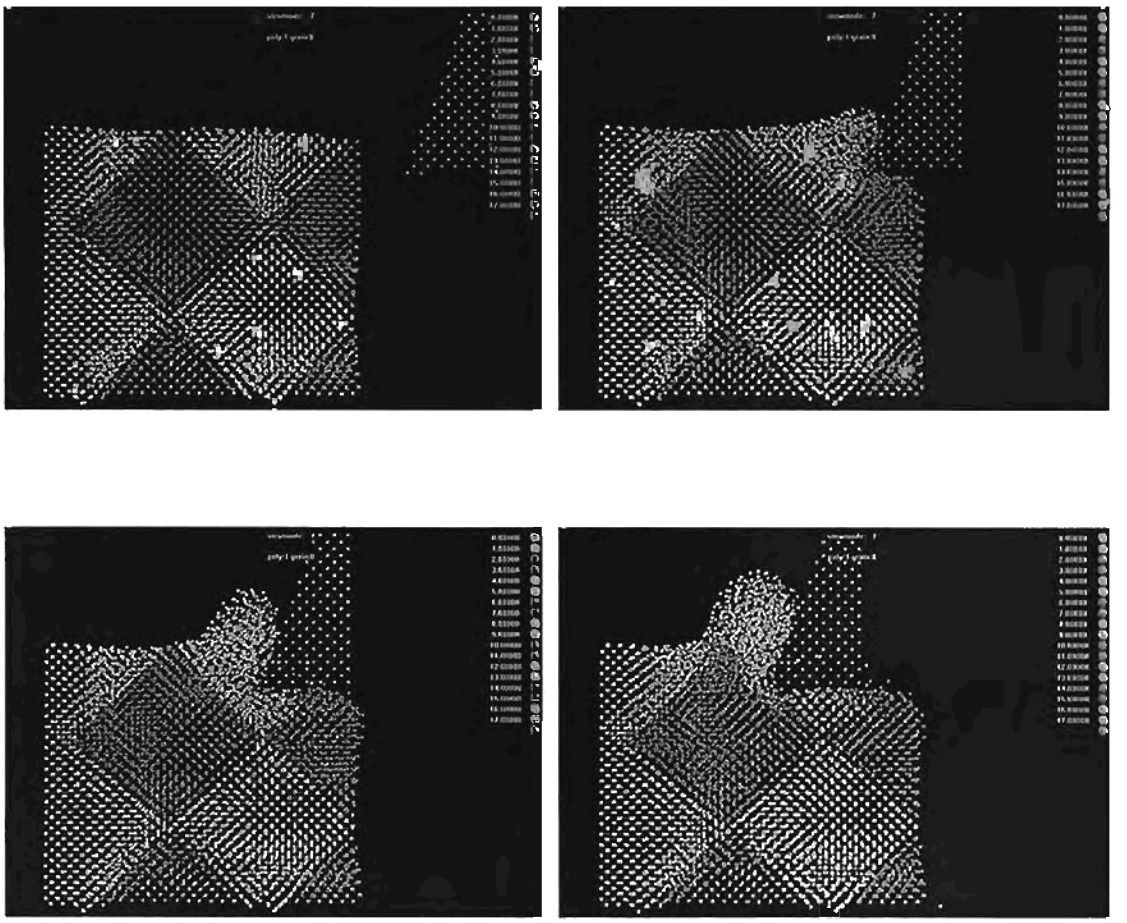


Figure 42a : Slides for Grain Test 8

MD simulation of nanometric cutting of Cu at various stages
Same as for 3 except "no mans land" is 1.0 instead of 0.5
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

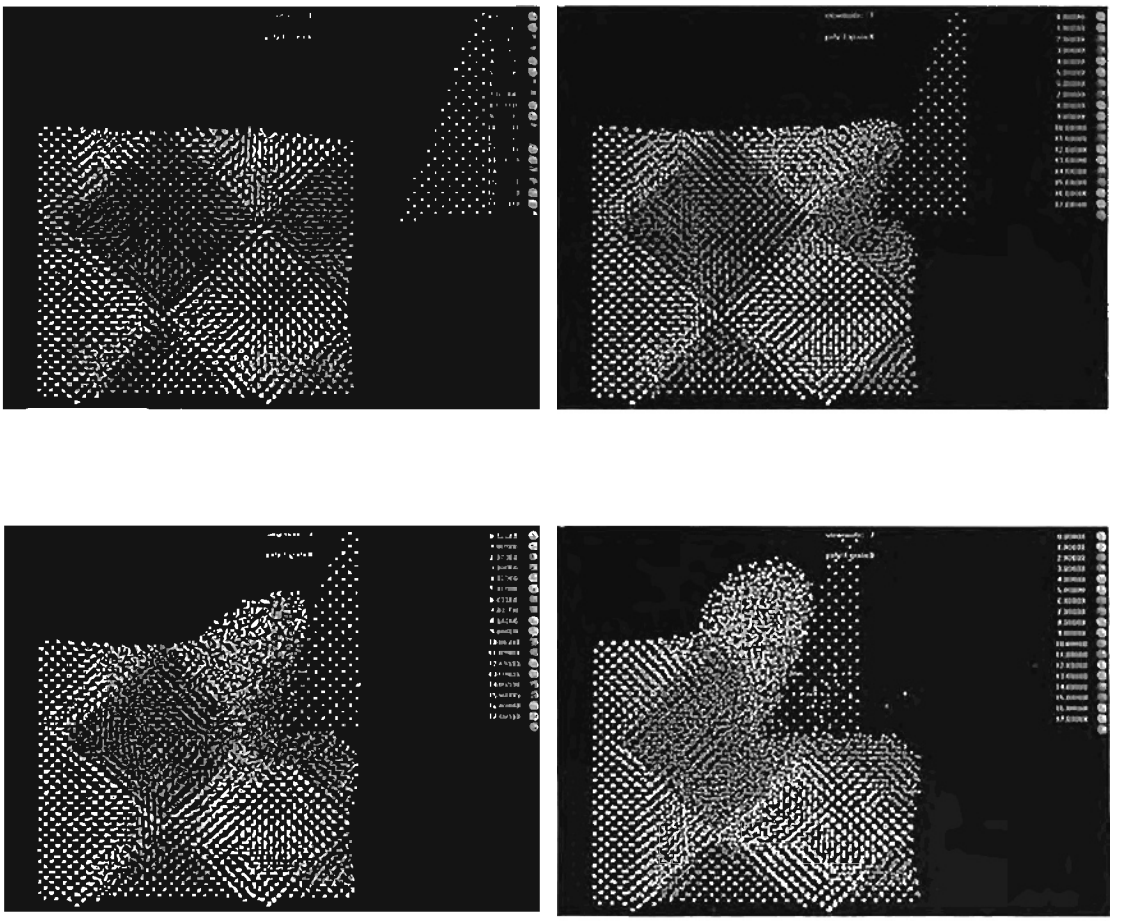


Figure 42b : Slides for Grain Test 8

MD simulation of nanometric cutting of Cu at various stages
Same as for 3 except "no mans land" is 1.0 instead of 0.5
Depth of cut : 2.2 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

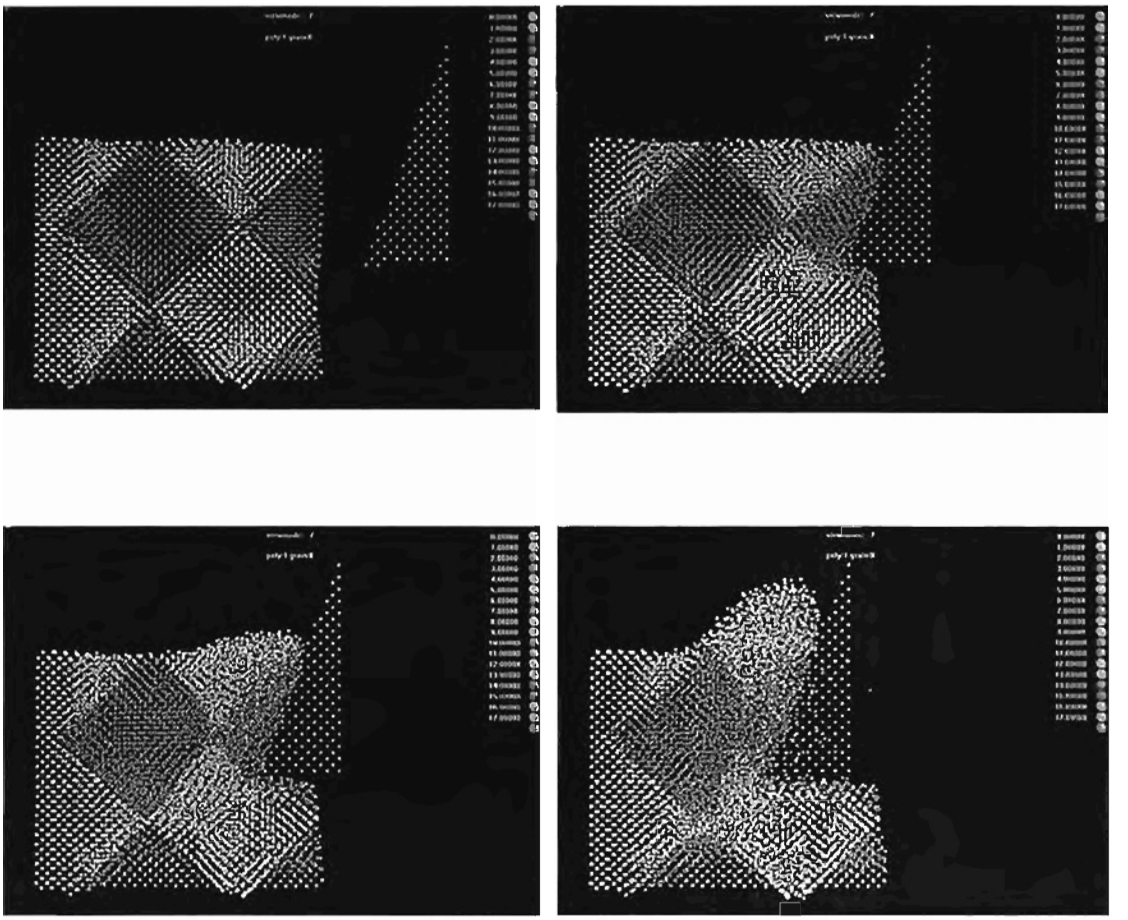


Figure 42c : Slides for Grain Test 8

MD simulation of nanometric cutting of Cu at various stages
Same as for 3 except "no mans land" is 1.0 instead of 0.5
Depth of cut : 3.3 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

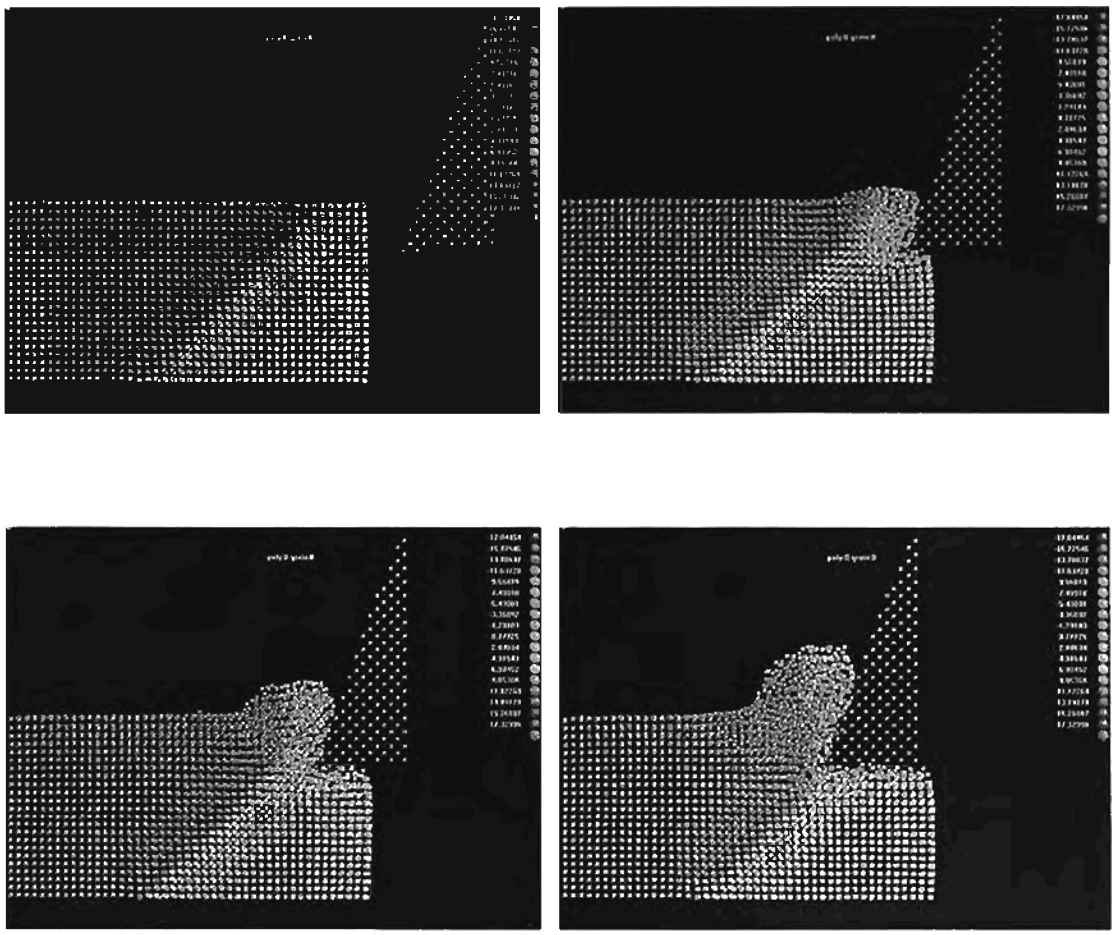


Figure 43 : Slides for Grain Test 9

MD simulation of nanometric cutting of Cu at various stages
Both grains are cubic orientation sharing a 45 degree boundary
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

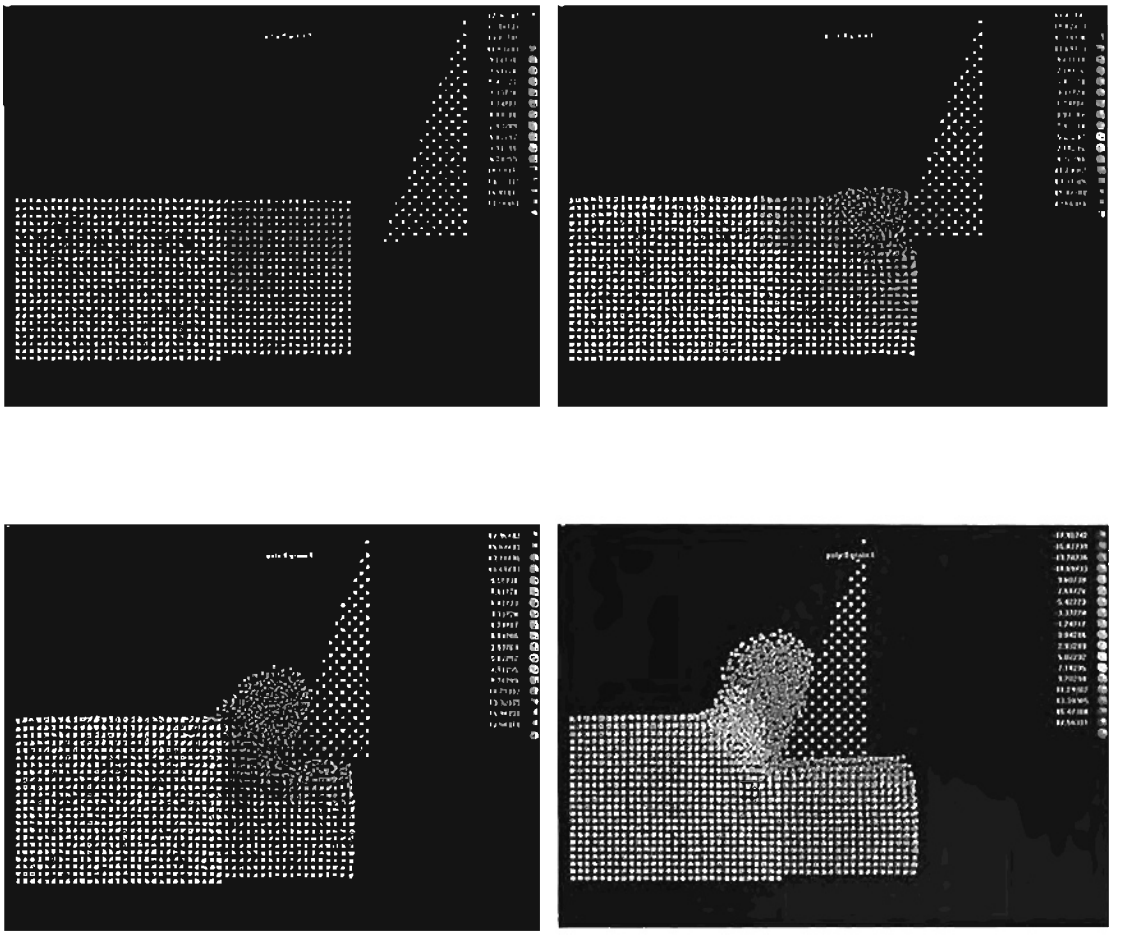
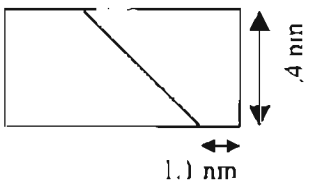


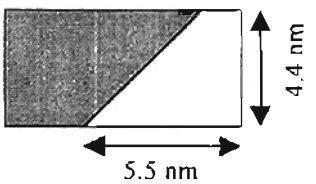
Figure 44 : Slides for Grain Test 10

MD simulation of nanometric cutting of Cu at various stages
Both grains are cubic orientation sharing a 90 degree boundary.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s



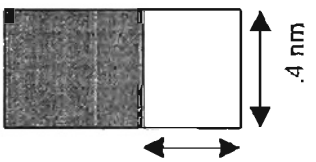
11) *Two grains of 0 degree orientation sharing a 135 degree boundary*

Exhibits stress lines parallel to grain boundary after relaxation. 45 degree shear angle ahead of tool. Roll is apparent, especially at grain boundary.



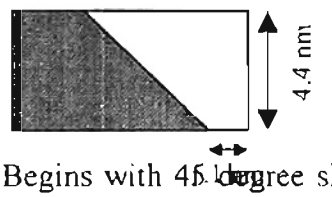
12) *Shaded grain with 45 degree orientation, unshaded 0 degree orientation sharing a 45 degree grain boundary*

Begins with 45 degree slip behavior, switches to 90 degree in 2nd grain. Rolling observed at grain boundary.



13) *Shaded grain with 45 degree orientation, unshaded 0 degree orientation sharing a 90 degree grain boundary*

Begins with 45 degree slip behavior, switches to 90 degree slip in 2nd grain. Rolling is observed at the grain boundary



14) *Shaded grain with 45 degree orientation, unshaded 0 degree orientation sharing a 135 degree grain boundary*

Begins with 45 degree slip behavior, changes to 90 degree slip in 2nd grain. Rolling is observed before and after boundary, but not right upon it.

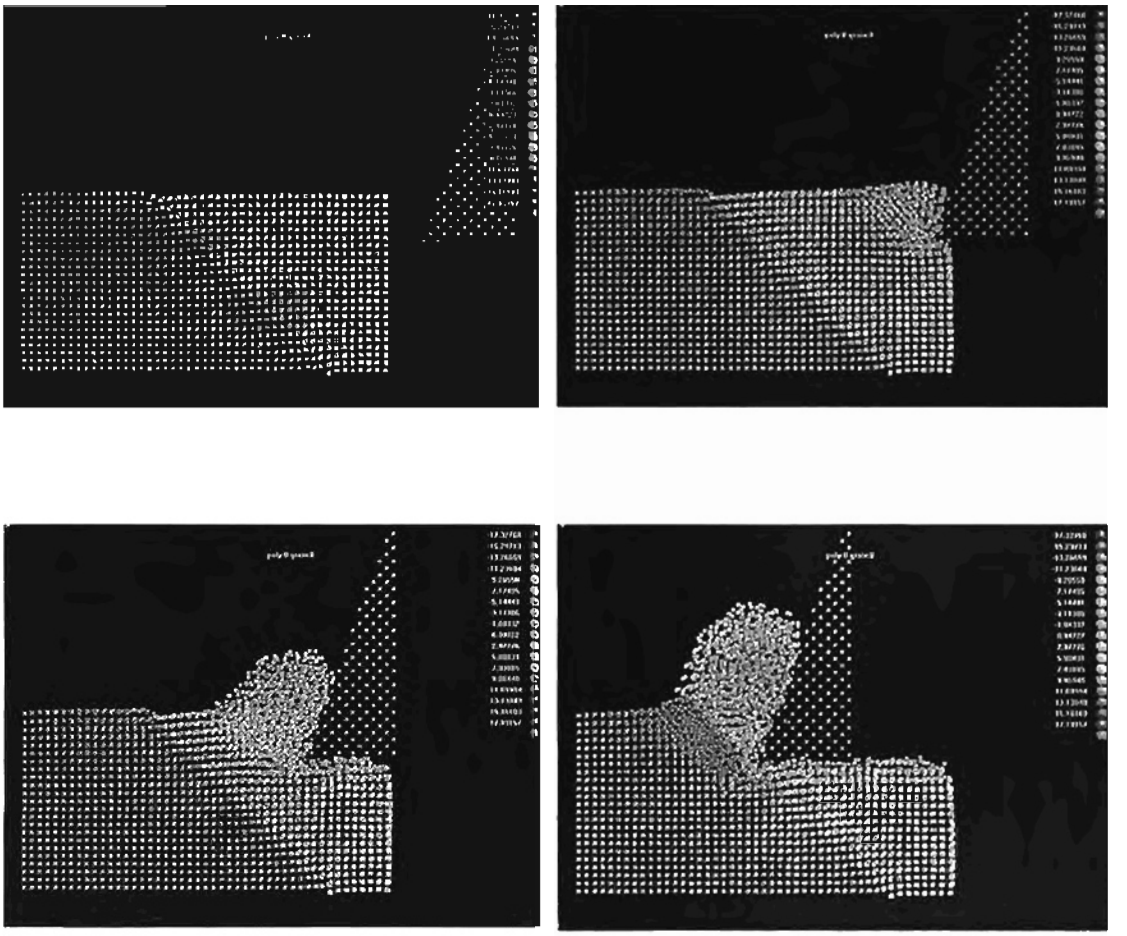


Figure 45 : Slides for Grain Test 11

MD simulation of nanometric cutting of Cu at various stages
Both grains are cubic orientation sharing a 135 degree boundary
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

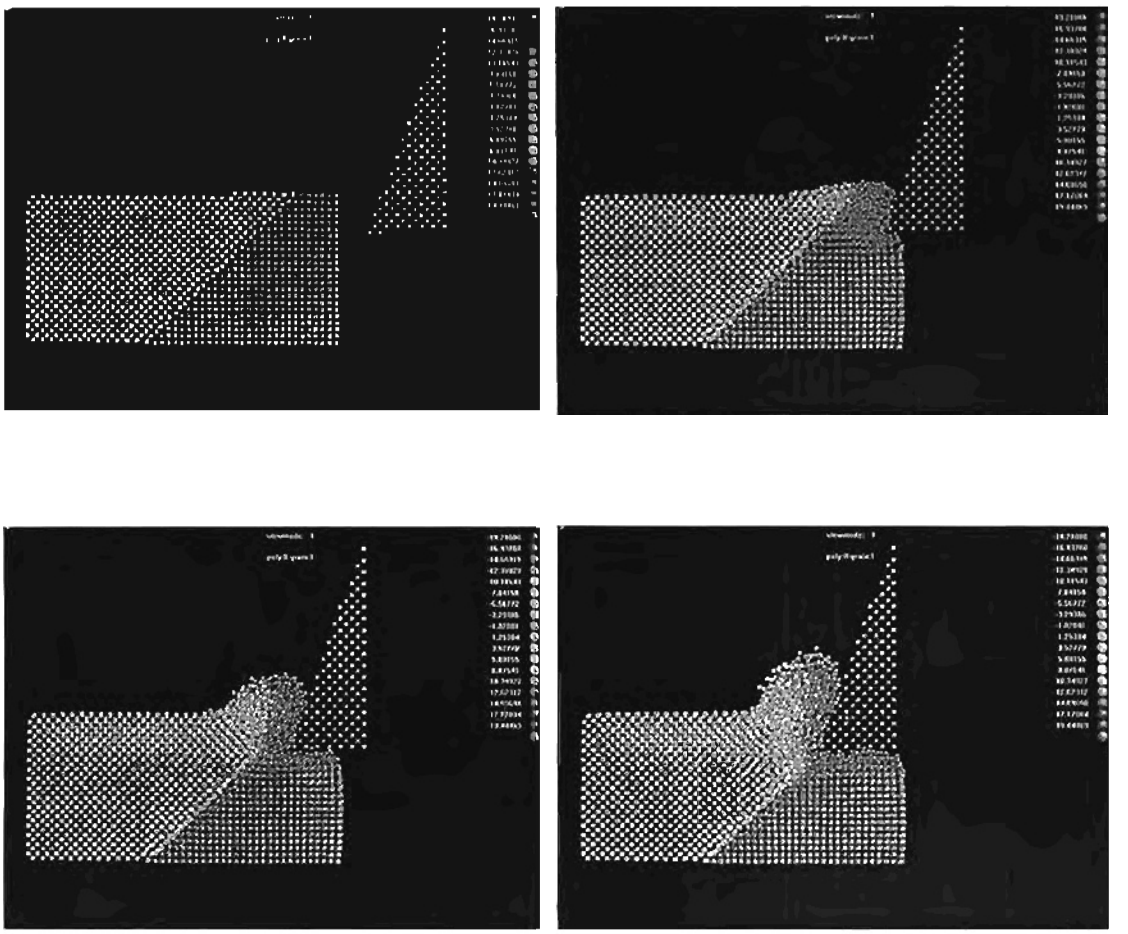


Figure 46 : Slides for Grain Test 12

MD simulation of nanometric cutting of Cu at various stages
Left most grain is 45 degree orientation, right grain is cubic.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

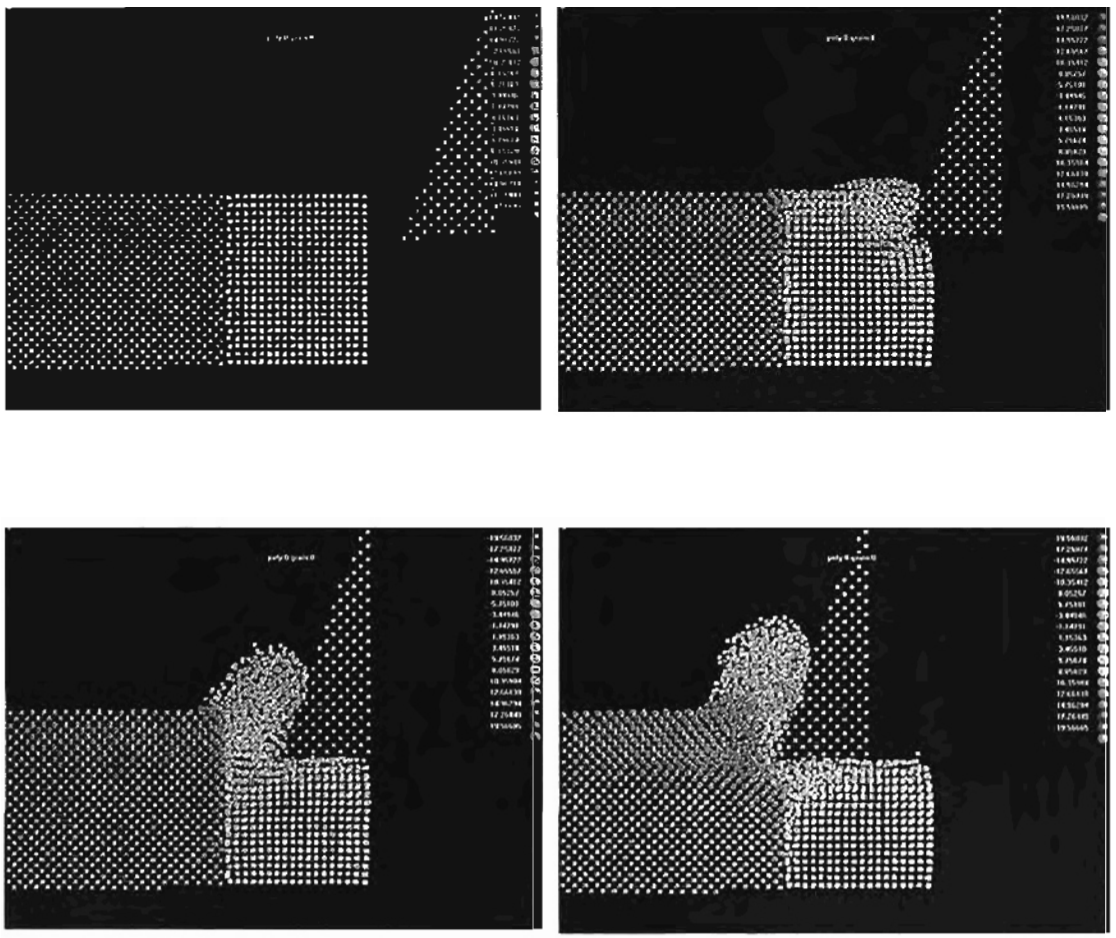


Figure 47 : Slides for Grain Test 13

MD simulation of nanometric cutting of Cu at various stages
Left grain is 45 degree orientation, right is cubic.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

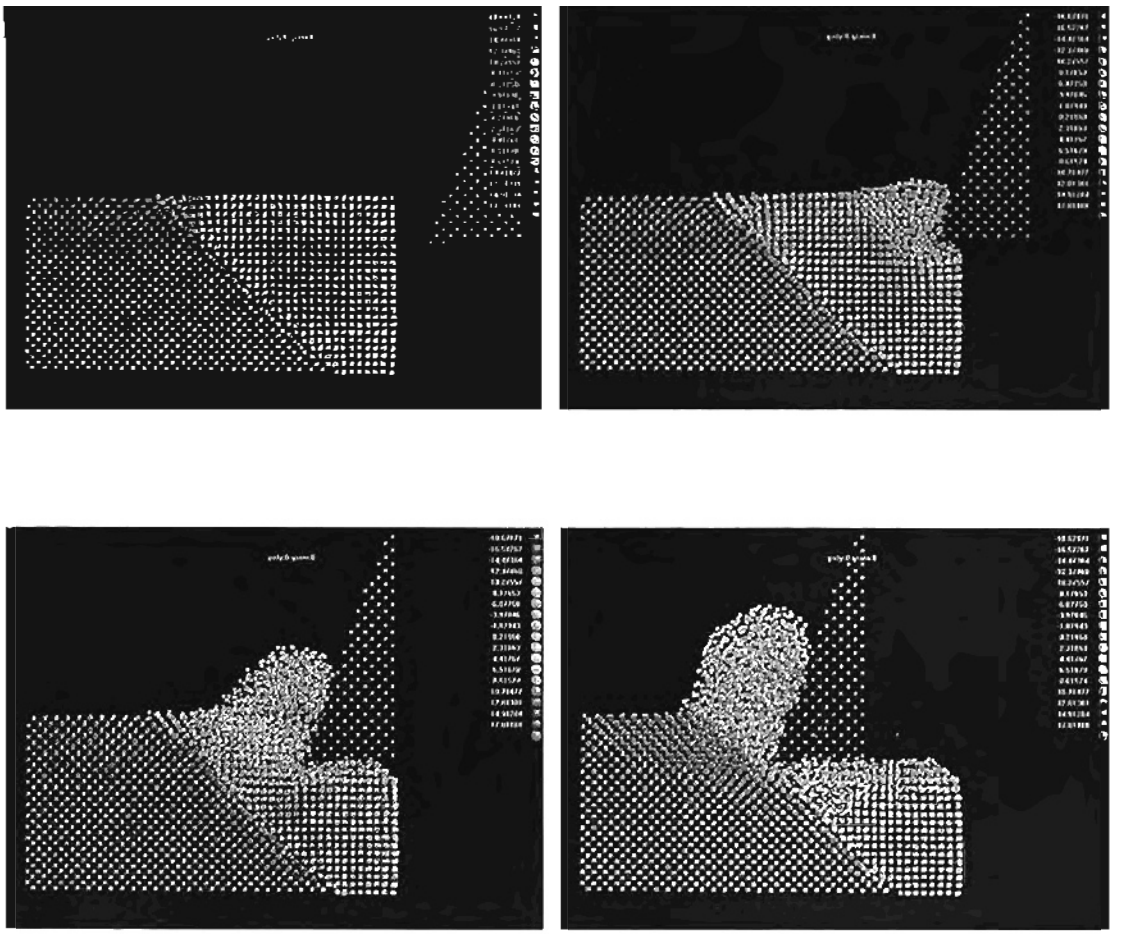
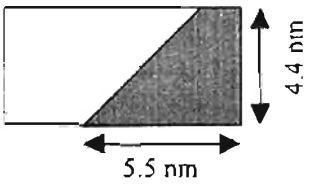


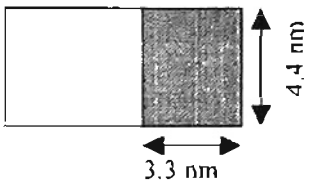
Figure 48 : Slides for Grain Test 14

MD simulation of nanometric cutting of Cu at various stages
Left grain is 45 degree orientation, right is cubic.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s



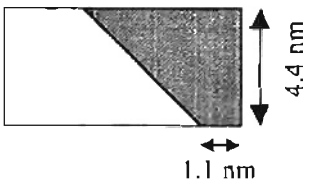
15) *Shaded grain with 45 degree orientation, unshaded 0 degree orientation sharing a 45 degree grain boundary*

45 degree stress lines are present after relaxation in left grain. Large amount of roll is observed near grain boundary and stress lines. Slips degrees in right grain, 0 degrees in left.



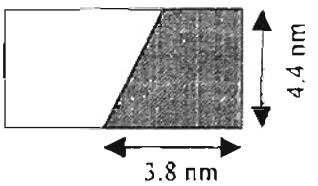
16) *Shaded grain with 45 degree orientation, unshaded 0 degree orientation sharing a 90 degree grain boundary*

90 degree slip in right grain, 45 degree slip in left grain.



17) *Shaded grain with 45 degree orientation, unshaded 0 degree orientation sharing a 135 degree grain boundary*

Slips 90 degrees all the way back to grain boundary in right grain, induces 45 degree stress lines, rolls near boundary, shifts to 45 degrees in left grain.



18) *Shaded grain with 67.5 degree orientation, unshaded 0 degree orientation sharing a 67.5 degree grain boundary*

Shaded grain exhibits 22.5 degree slip, switches to 45 degree slip in left grain. Leaves a jumbled region near where boundary was after tool passes.

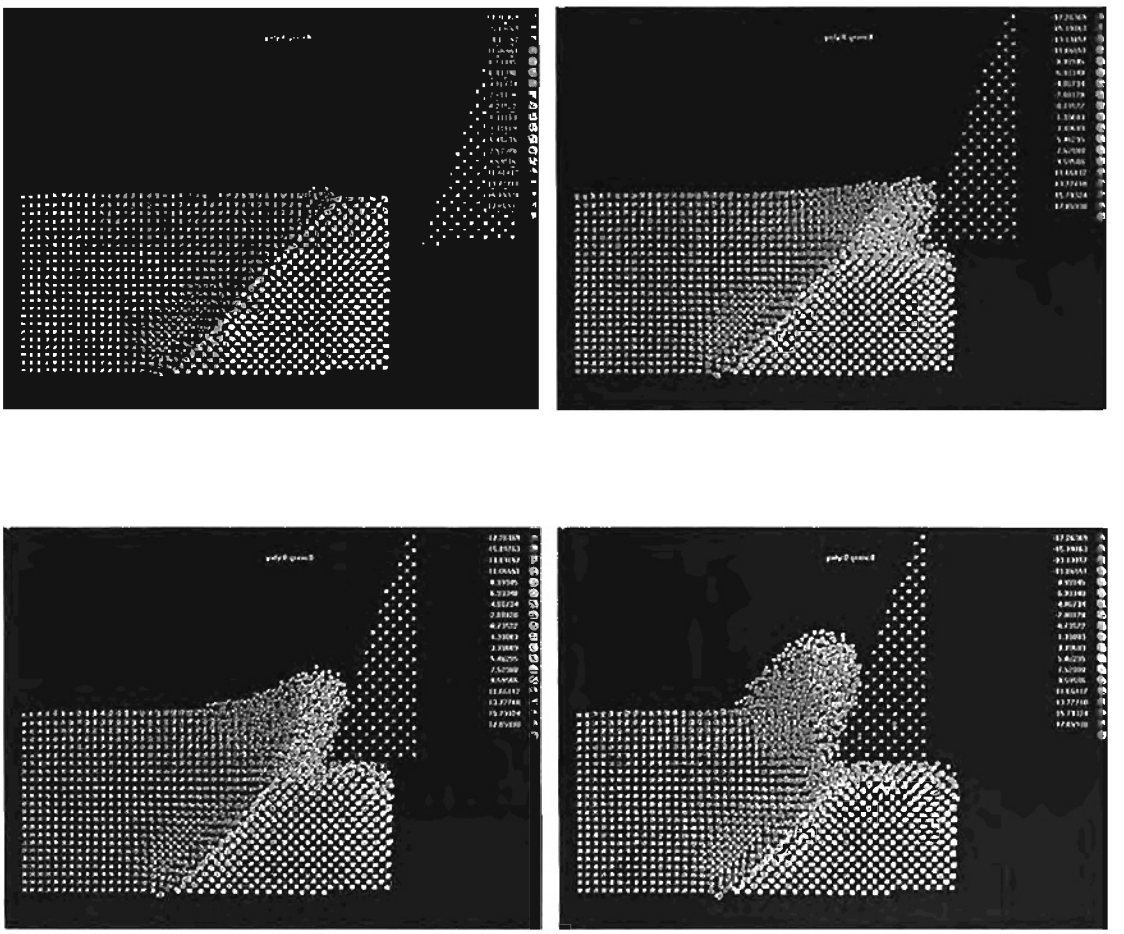


Figure 49 : Slides for Grain Test 15

MD simulation of nanometric cutting of Cu at various stages
Left grain is cubic, right is 45 degree orientation.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

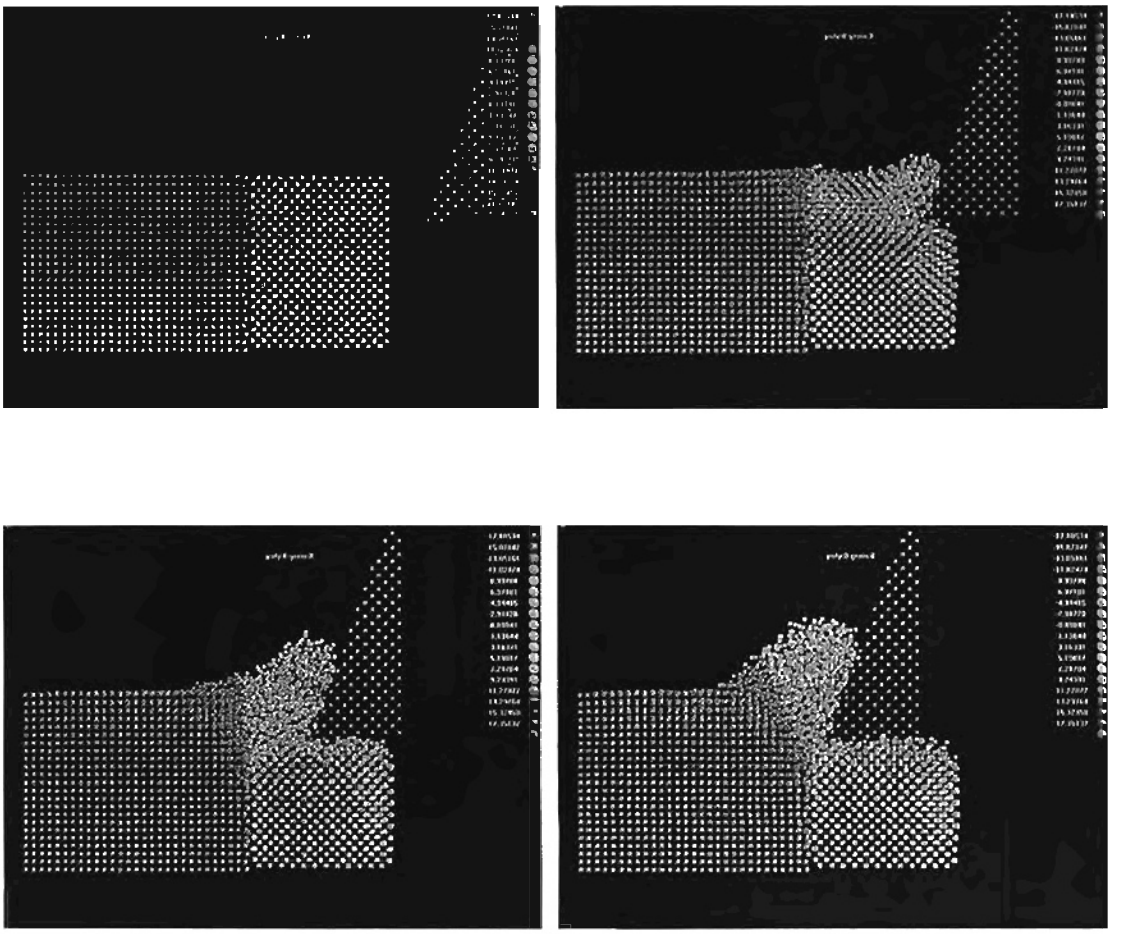


Figure 50 : Slides for Grain Test 16

MD simulation of nanometric cutting of Cu at various stages
Left grain is cubic, right is 45 degree orientation.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

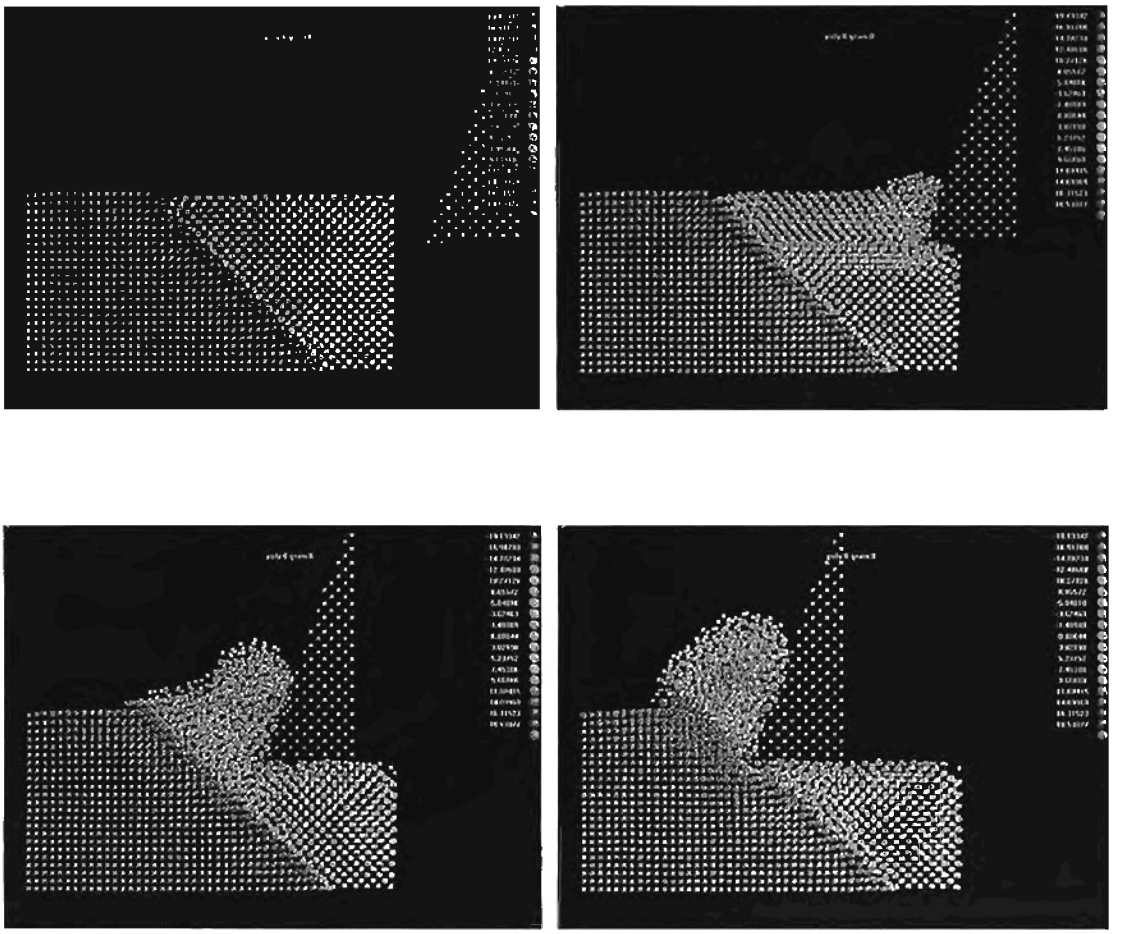


Figure 51 : Slides for Grain Test 17

MD simulation of nanometric cutting of Cu at various stages
Left grain is cubic, right is 45 degree orientation.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

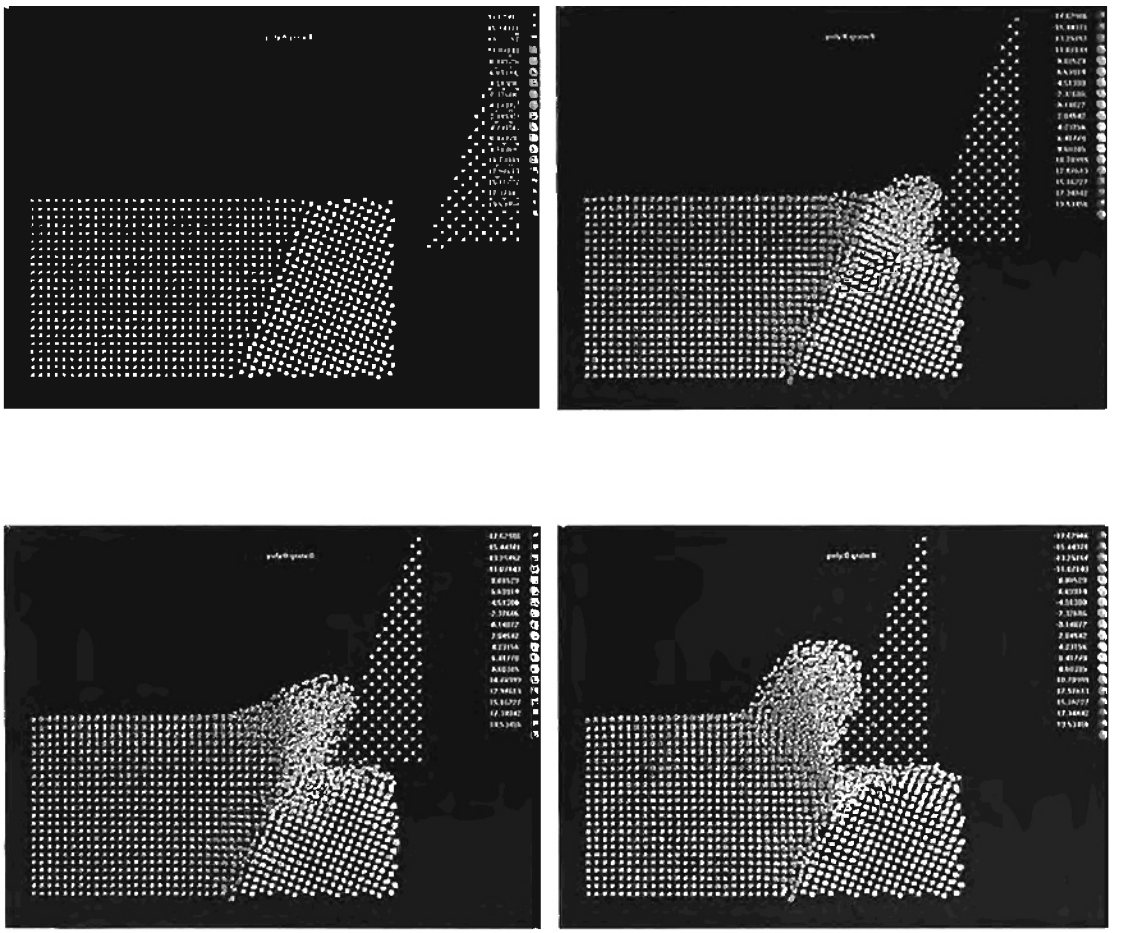
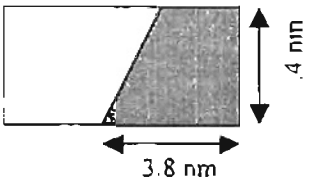


Figure 52 : Slides for Grain Test 18

MD simulation of nanometric cutting of Cu at various stages
Left grain is cubic, right is 67.5 degree orientation.
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

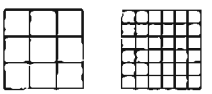


19) *Shaded grain with 67.5 degree orientation, unshaded 0 degree orientation sharing a 67.5 degree grain boundary and with a "no-man's land" of 0.75 Å*

22.5 slip in shaded, 45 slip in unshaded, leaves a "transition region" near where grain boundary was after machining which goes from 67.5 to 90

Force comparisons: (these are generalizations across multiple depths of cut. See actual diagrams that follow for individual behavior)

(1) vs (2) : Similar cutting force magnitudes (about 100 eV/Å for depth of cut 2.2 nm) , more thrust force for (2) (. Thrust force displayed a reversal in direction (more force down on the tool) with increasing depth of cut for (1), (2) displayed this as well in the first half of the workpiece. Cutting force is more stable for (2). The cutting force gained in magnitude to its maximum extent after the tool tip passed through the first grain boundary (at time = 640 atu) for (1), whereas it peaked at the first grain boundary (time=420 atu) for (2).



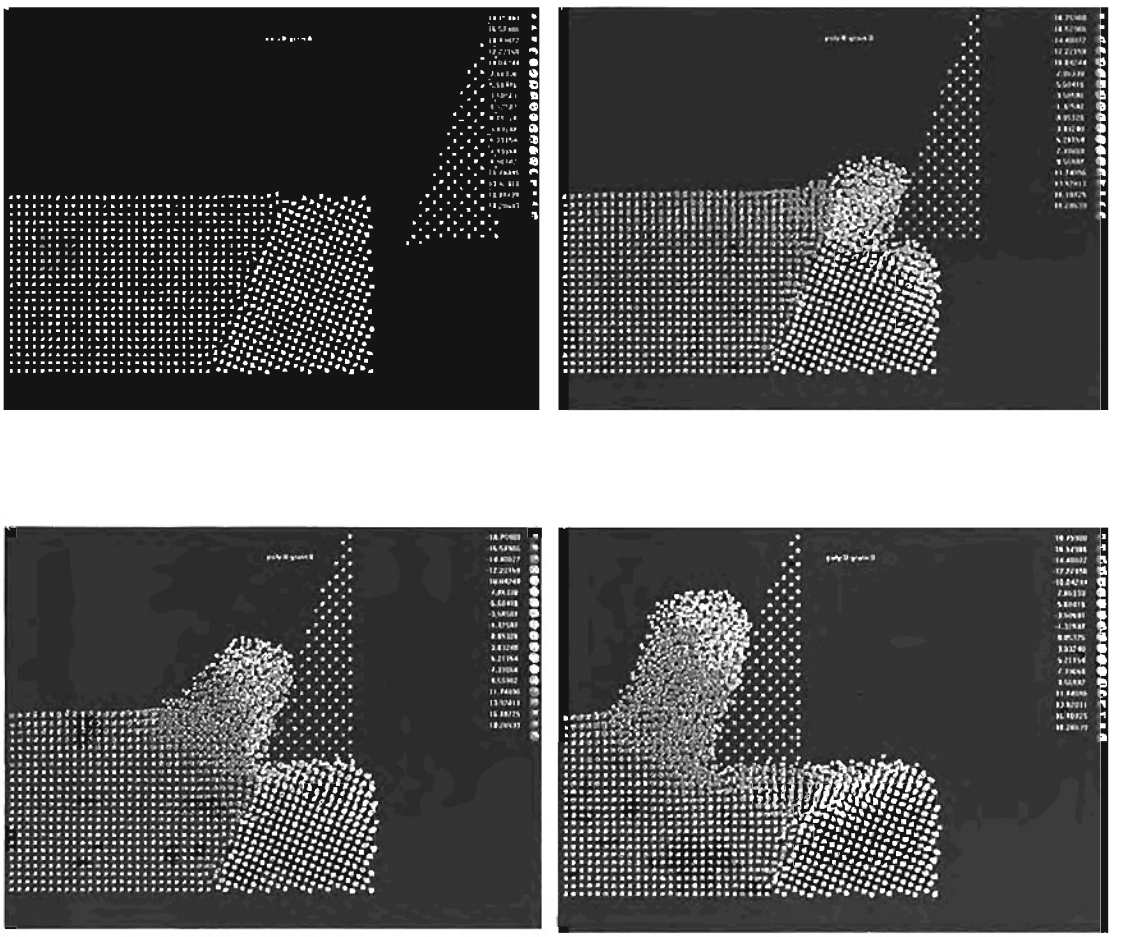


Figure 53 : Slides for Grain Test 19

MD simulation of nanometric cutting of Cu at various stages
Same as 18 except "no man's land" is 1.5 angstroms not 2.0
Depth of cut : 1.1 nm
Tool rake angle : 20 degrees
Tool clearance angle : 5 degrees
Cutting Speed : 500 m/s

FIGURES 54a-c : RUN 1 FORCES

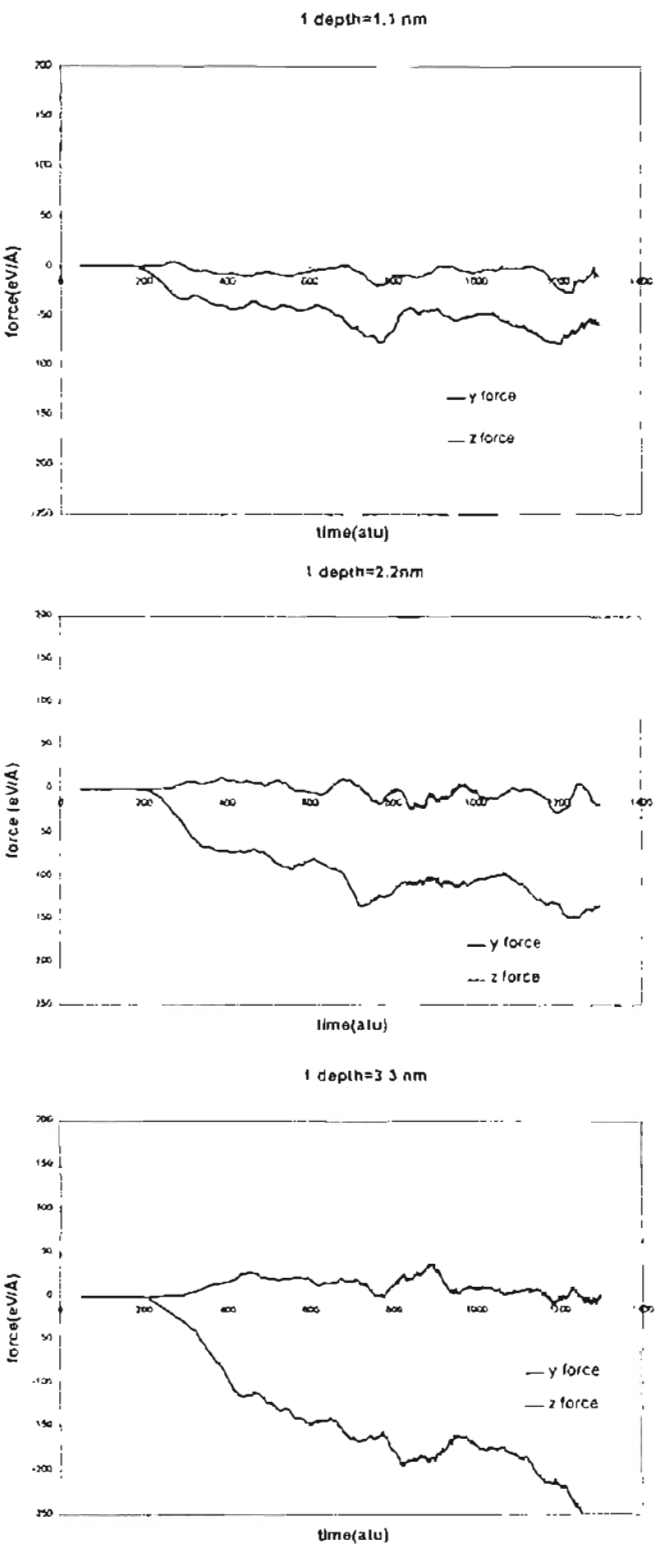
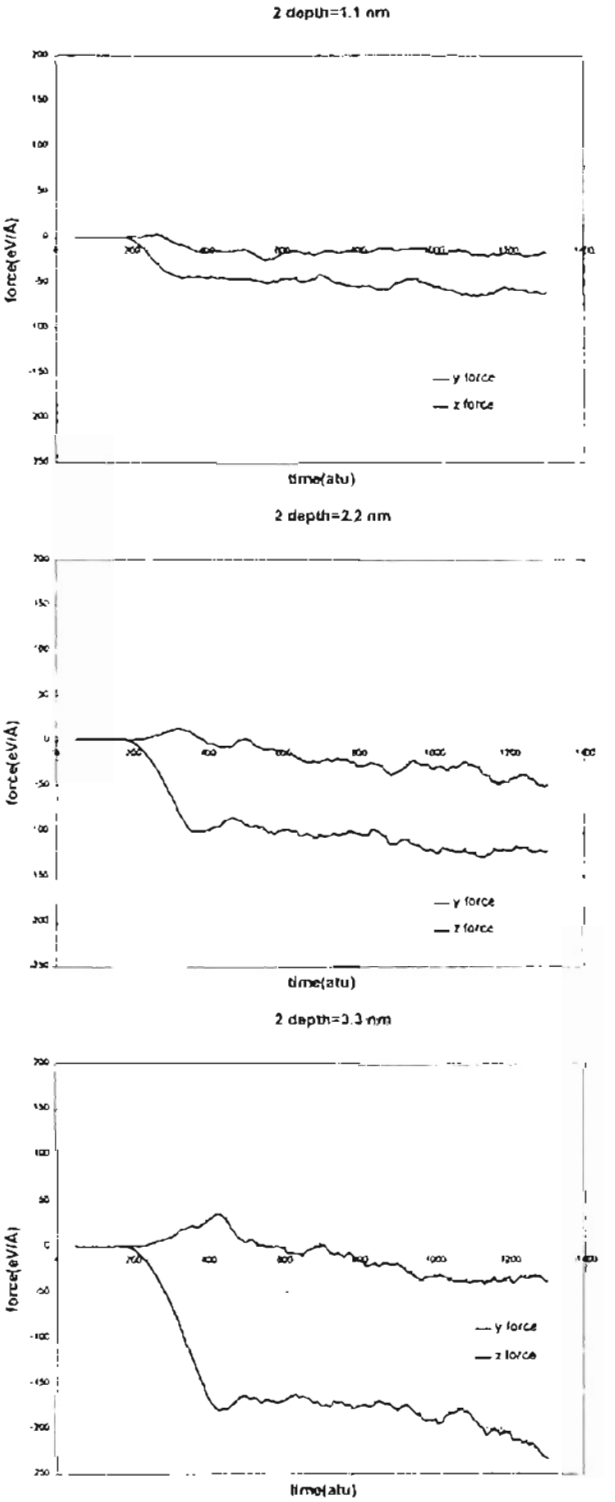


FIGURE 55a-c : RUN 2 FORCES




(1) vs (3) : Similar magnitudes (cutting force about 100 eV/Å and thrust force near 0 eV/Å for depth of cut 2.2 nm), stepped entry and spiky for (1), sharper entry and more stable for (3). Grain boundary comparisons are somewhat irrelevant here due to the diagonal nature of the boundaries in (3) (the tool continually crosses boundaries in a different fashion than for the vertical boundaries in (1)).


(1) vs (5) : (Recall that shaded grains are at 45 degrees orientation) Similar magnitudes (cutting force about 100 eV/Å, thrust force near 0 eV/Å for depth of cut 2.2 nm). Both (1) and (5) display a local minimum in cutting force when crossing the first grain boundary (at time = 640 atu)


(2) vs (3) : Both display a reversal in direction of thrust force with increasing depth of cut (towards more force downward on the tool). Sharper entry derivative on cutting force in (3) for the smallest depth of cut (1.1nm), otherwise similar magnitudes (cutting force about 100 eV/Å, thrust force near 0 eV/Å for depth of cut 2.2 nm)


(2) vs (4) : Similar magnitudes (about 100 eV/Å cutting force, near 0 eV/Å thrust force for depth of cut 2.2 nm), cutting forces are much more stable for (2) (there is a cutting force variation of about +/- 20 eV/Å for (4) during the middle of the simulation whereas it is only about +/- 5 eV/Å for (2))

FIGURE 56a-c : RUN 3 FORCES

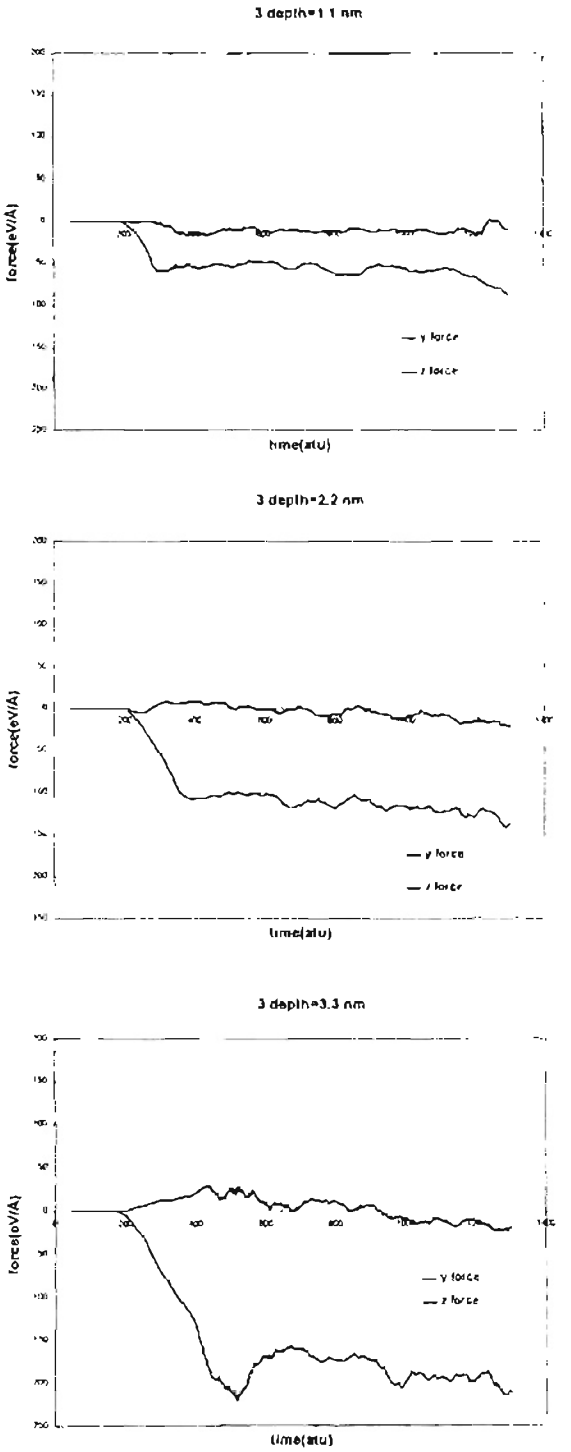


FIGURE 57a-c : RUN 4 FORCES

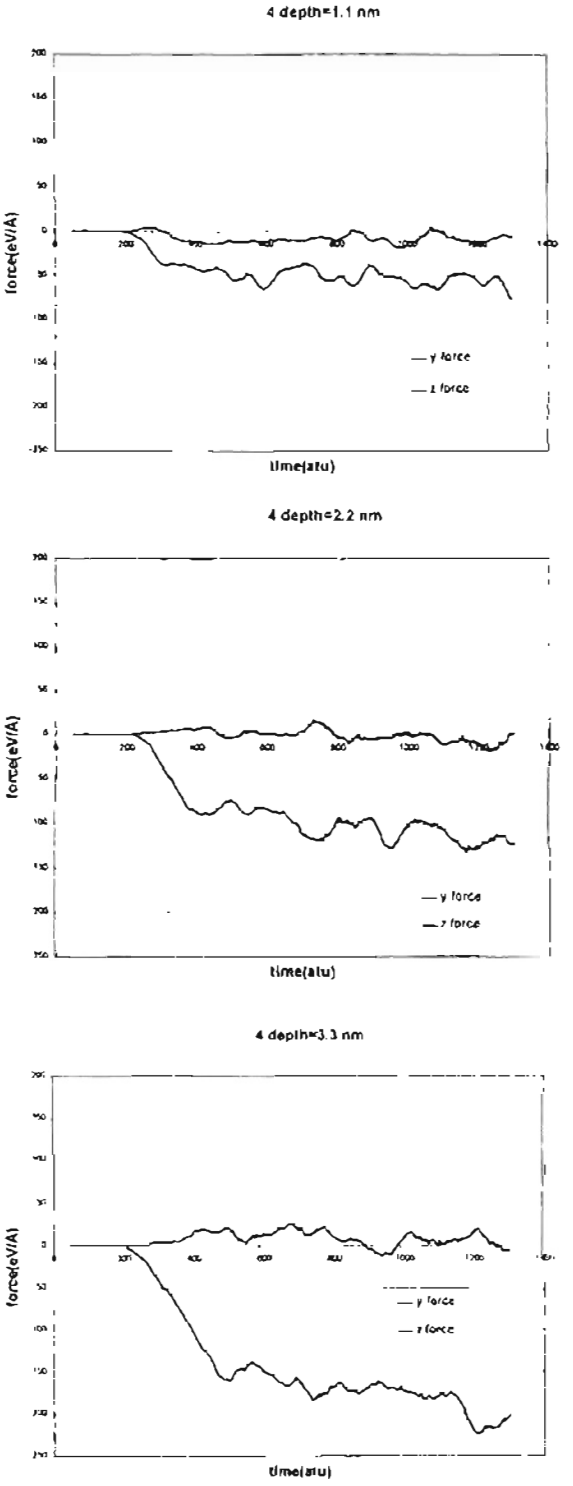
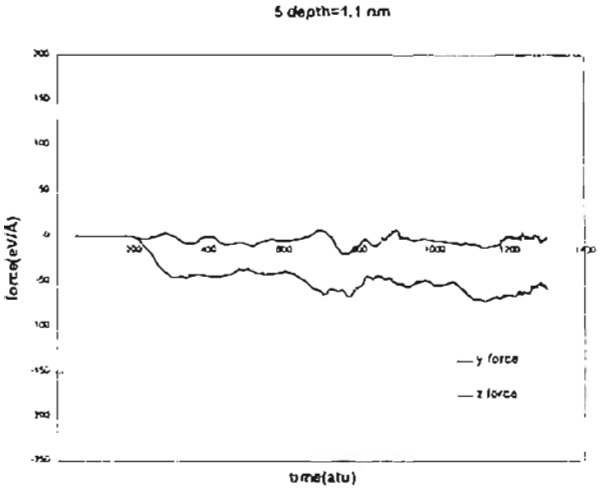
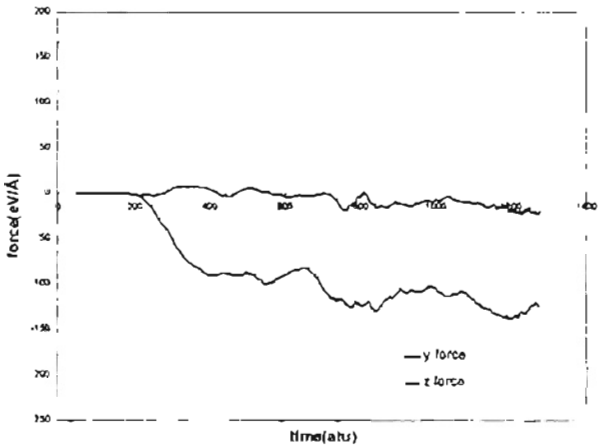
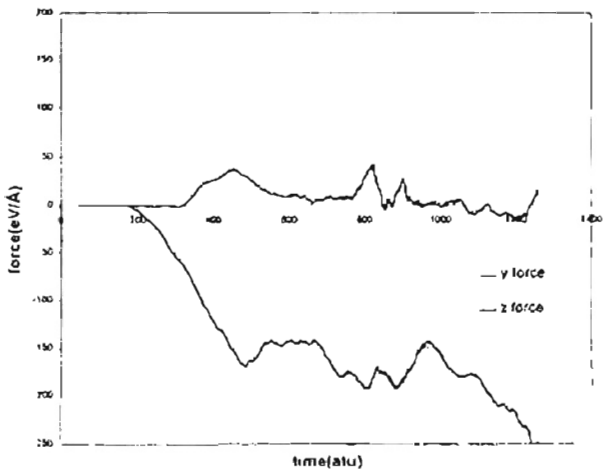


FIGURE 58a-c : RUN 5 FORCES







  (2) vs (6) : Similar magnitudes (about 100 eV/Å cutting force, near 0 eV/Å thrust force for depth of cut 2.2 nm) . Whereas maximum forces are reached in (2) at or prior to crossing of the first grain boundary, the maximum forces in (6) tend to be centered in the middle of the second grain.

  (3) vs (7) : (Recall here that shaded grains are of zero degree orientation) Higher initial cutting force for (3) (peaks at about 100 eV/Å at time = 375 atu. (7) only reaches about 75 eV/Å at this time but eventually even up with (3) at time = 600 atu) more small spikes for (7)

  (3) vs (8) : Similar magnitudes (cutting force about 100 eV/Å, thrust force near 0 eV/Å for depth of cut 2.2 nm), sharper entry for (3) then stable (peaks at time=375 atu for depth of cut 2.2 nm), shallower entry then spiky for (8) (peaks at time=575 atu for depth of cut 2.2 nm). "Anti-peaks" are present in (8), where thrust force and cutting force spike in opposite directions (time = 425 atu for depth of cut 1.1 nm and time = 575 atu for depth of cut 2.2 nm)

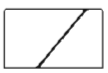
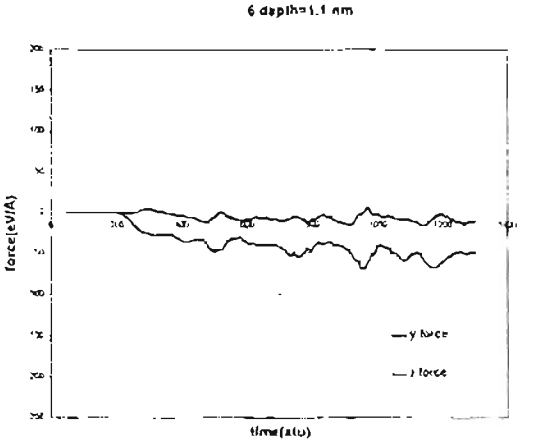
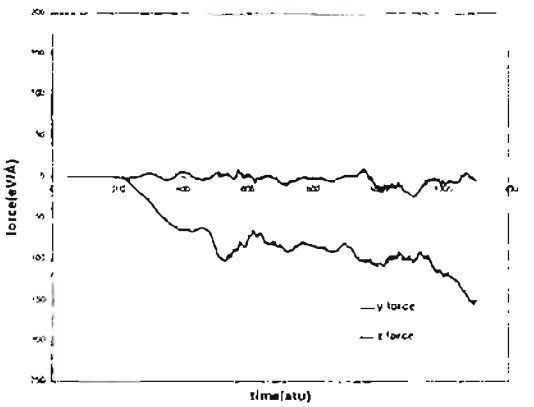
  (9) vs (15) : (Recall that shaded grain is at 45 degrees orientation) Very large "anti-peak" behavior in both when the tool tip crosses the grain boundary (time = 600 atu), but more pronounced in (9) (spike of about 20 eV/Å for (15), 40 eV/Å for (9)). (15) has higher initial cutting force (peaks at about 50 eV/V, whereas (9) peaks early at about 40 eV/Å)

FIGURE 59a-c : RUN 6 FORCES





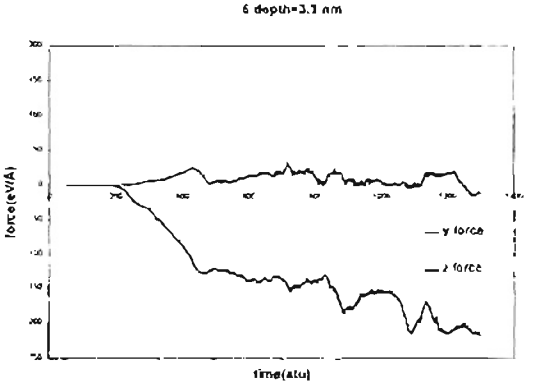


FIGURE 60a-c : RUN 7 FORCES

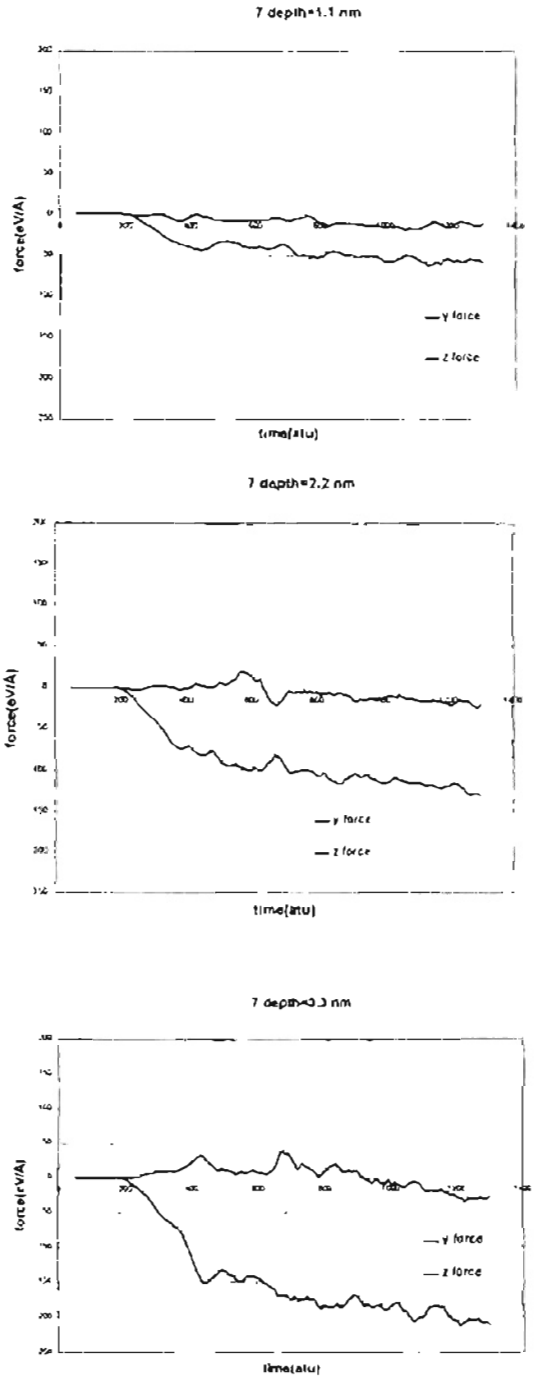
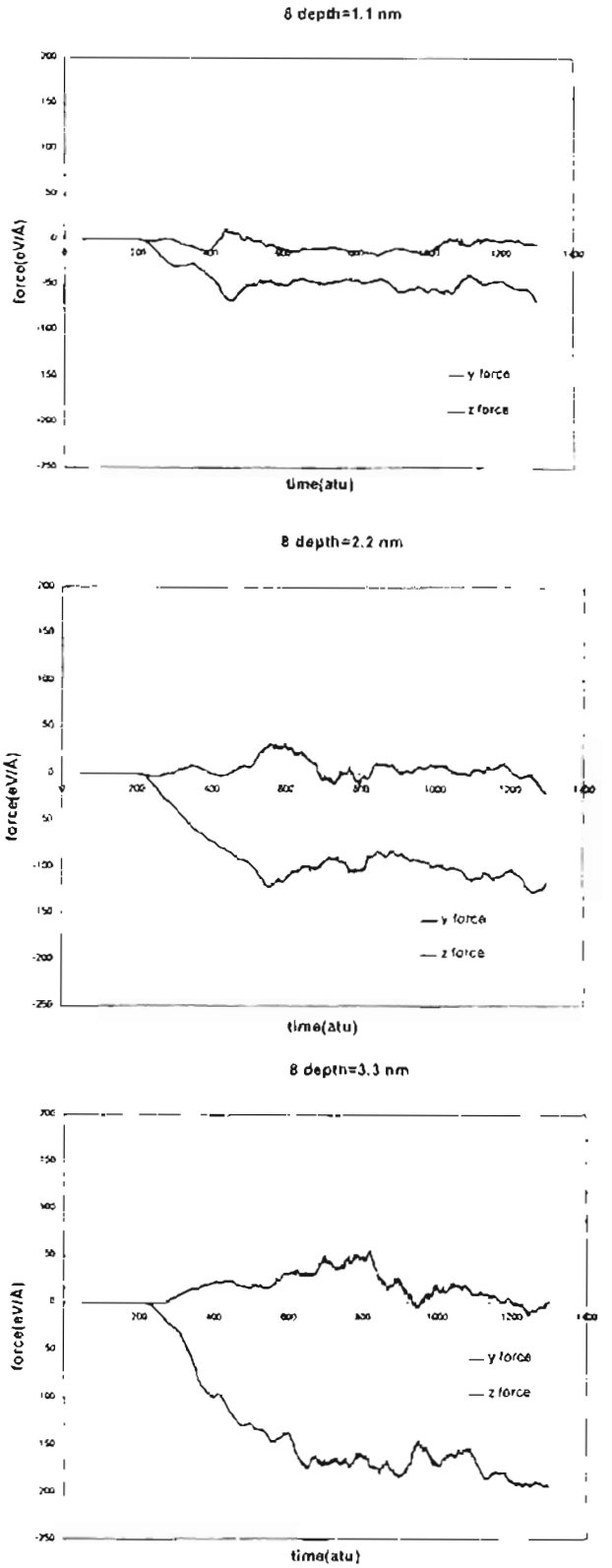
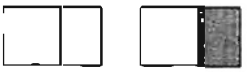
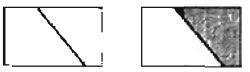
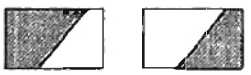


FIGURE 61a-c : RUN 8 FORCES



 (10) vs (16) : (shaded grain at 45 degrees orientation) Similar behavior with slightly more thrust force for (10) (cutting force about 50 eV/Å, thrust force around 20 eV/Å for (10), 15 eV/Å for (16)) No significant behavior near the grain boundary.

 (11) vs (17) : (shaded grain at 45 degrees orientation) Similar cutting force behavior, with more thrust force in (11) (cutting force about 50 eV/Å, thrust force about 20 eV/Å for (11), averaging 10 eV/Å for (17)). Cutting force initially peaks out higher for (17). Both thrust and cutting forces begin to increase past the grain boundary for (17), whereas no real definable behavior for (11).

 (12) vs (15) : (shaded grains at 45 degree orientation) Distinct anti-peak present in (15) when tool crosses grain boundary at time=600, no discernable behavior at boundary for (12). Lower thrust force in (15) (thrust force averages about 15 eV/Å for (15), 20eV/Å+ for (12)), with similar cutting forces for both (initially about 50 eV/Å progressing to 60 eV/Å over time)

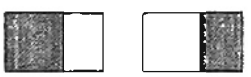
 (13) vs (16) : Similar behavior (cutting force averaging about 50 eV/Å before the grain boundary, thrust forces about 15 eV/Å). Cutting force for (13) increases to higher value as the tool tip nears the grain boundary (time = 825 atu cutting force for (13) is 60 eV/Å, (16) is 50 eV/Å)) and then both cutting and thrust forces increase for each simulation past the grain boundary.

FIGURE 62a-c FORCES FOR RUNS 9-11

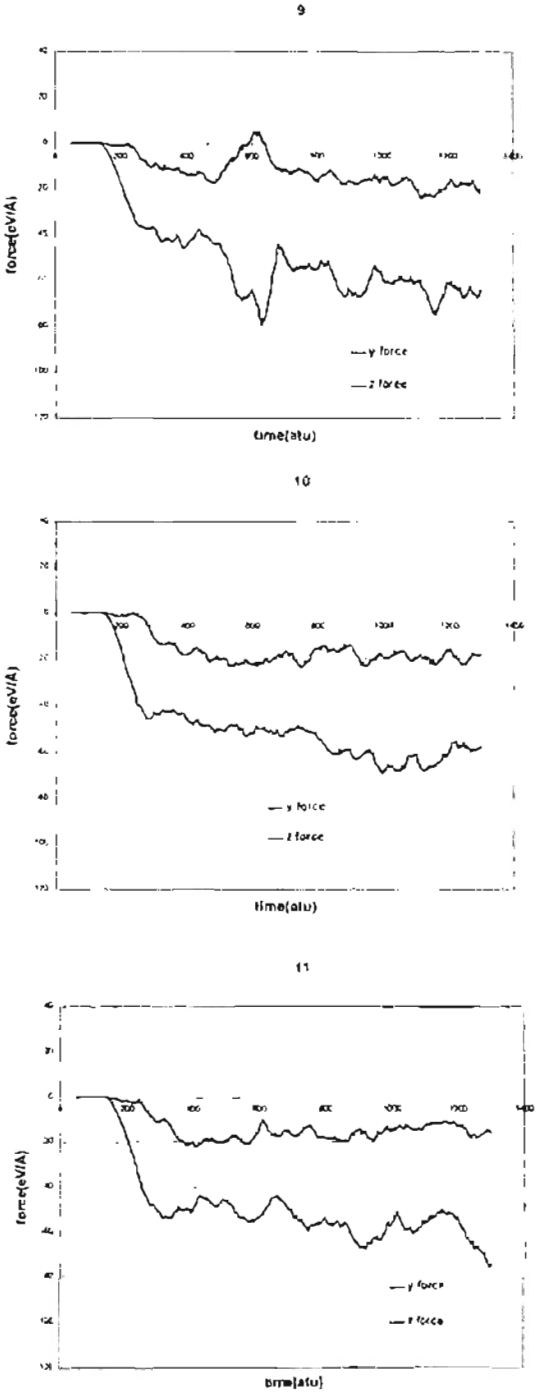


FIGURE 63a-c : FORCES FOR RUNS 12-14

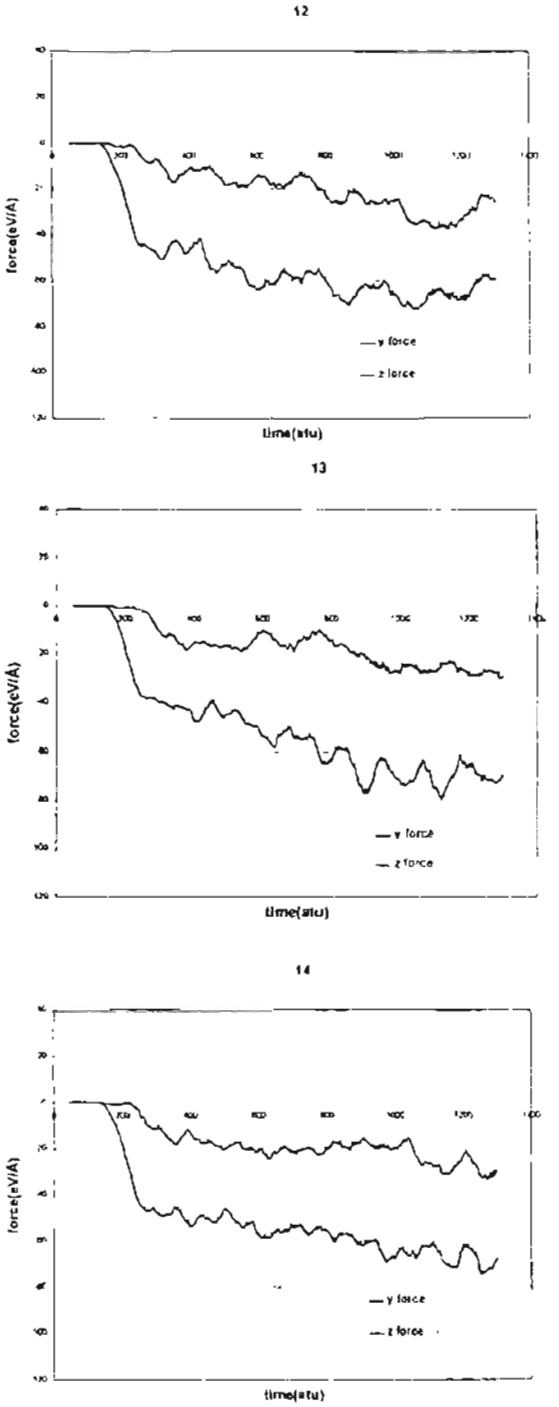


FIGURE 64a-c : FORCES FOR RUNS 15-17

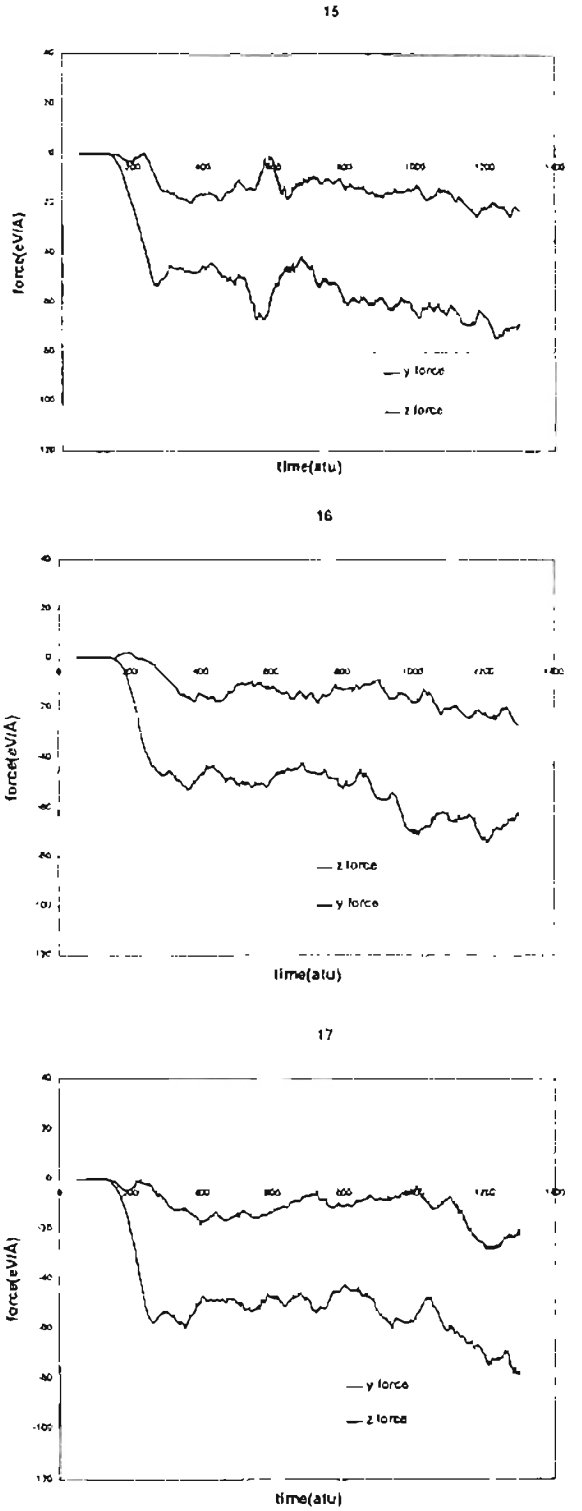
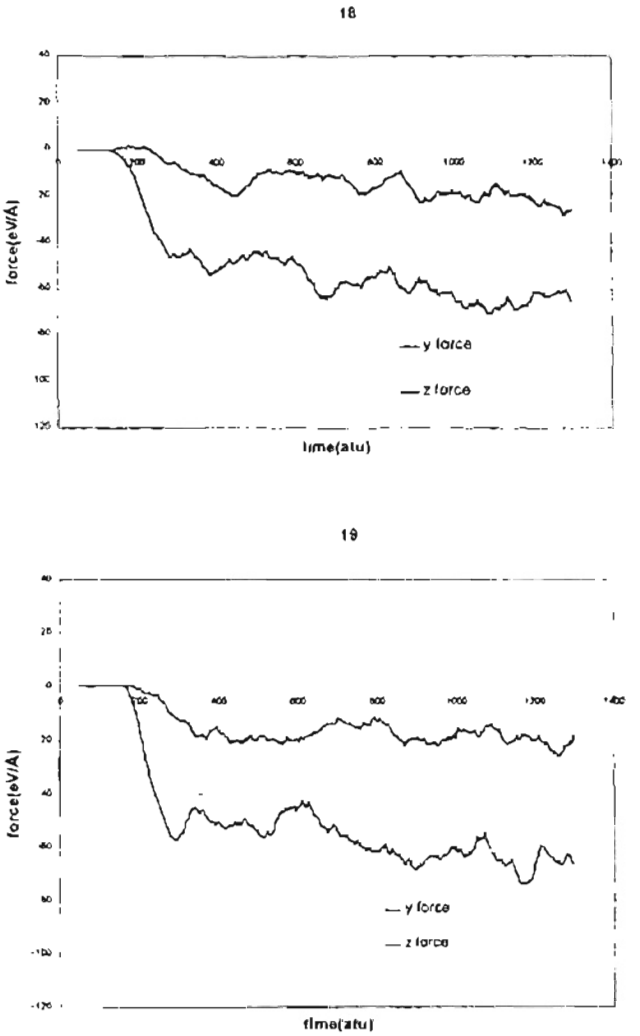


FIGURE 65a-b : FORCES FOR RUNS 18-19



  (14) vs (17) : (shaded grains at 45 degree orientation) Lower thrust force in (17) (thrust force about 20 eV/Å for (120, averaging near 10 eV/Å for (17)), greater initial cutting force in (17) ((17) spikes to 60 eV/Å at time=250 atu, (14) is only at 50 eV/Å) but (14) eventually surpasses due to its steady increase. Once the grain boundary is crossed however, (17)'s cutting and thrust forces increase rapidly whereas (14)'s display only a minor increase.

  (18) vs (19) : (shaded grains at 67.5 degree orientation, parallel to boundary) Thrust force more stable for (19) before the tool reaches the grain boundary (a somewhat steady value of 20 eV/Å for the first 600 atu, whereas thrust force varies between 20 and 15 eV/Å for (18) during this time). Cutting force is similar (averaging about 50 eV/Å) with a higher initial peak value (60 eV/Å vs. 50 eV/Å) for (18). After crossing the boundary forces reach similar behavior, although there is more of a gap between thrust and cutting force at the grain boundary for (18) than for (19).

Note: the unit of time on the force graphs is in atu, or "atomic time units". These were the basic time units utilized in the simulation, and 1 atu is on the order of $1 * 10^{-14}$ sec (about 10 fs). Also note that the y force is horizontal (cutting) force and z force is vertical (thrust) force. Also, magnitudes of the forces are for reaction forces, therefore the actual applied force upon the tool is opposite in direction but of same magnitude.

7.4. Discussion of Results

The results of the various nanograin simulations lead to the following conclusions:

1. By decreasing grain size and with grains all of similar orientation (i.e. no grain boundary angle) the behavior becomes more homogenous. Dislocations are observed to progress through multiple grains in a very pronounced manner, implying that the individual grains do not distort deformation behavior across their boundaries as much as for larger grains.

2. For grains with high neighboring grain angles (specifically 45 degrees) and decreasing grain size there is a similar loss of "grain individuality". In the larger grain sizes, the deformation behavior can be seen to change direction within each grain to that particular grains' preferred direction of slip, while in the small grains there is less distinction. This perhaps explained the "rolling" behavior discussed next:

3. When crossing grain boundaries, especially between two grains of dissimilar orientation, a "rolling" effect is observed. This can be characterized by the atoms of each respective grain being displaced by the tool, influenced by all the neighboring grains and trying to reorient themselves to a new position. The more grain boundaries nearby, the more pronounced the roll. The effect also continued somewhat into a grain after crossing a boundary, and so in the smaller grain sizes a near constant roll was observed.

From the above statements, it appears that for very small grain sizes the grains begin to lose their individual identity and consequently their influence upon deformation. They are simply not large enough to allow for a "stabilization" of deformation into their preferred direction. The magnitudes of force do not differ substantially between similar arrangements of high angle boundaries and zero angle boundaries, but there are some trends which seem to occur:

a. The arrangements with alternating grain orientations (and thus "high angle" grains) produce more of a periodic force (characterized by small spikes) than the non-alternating grains.

b. For non-alternating grain orientations, those with smaller grain size displayed more stable force behavior

c. Two situations produced a large "anti-peak" at the grain boundary: experiments (9) and (15). This anti-peak was typified by a drop in thrust force with a corresponding increase in cutting force.

Trend A is likely to be explained by generality number 3 (the "rolling effect" mentioned earlier). Alternating grains would continuously need to "roll" and thus produce a more periodic effect. Trend B is explained by generality number 1. The more homogenous slip observed in the smaller grain sizes would explain the more stable force readings. Trend C is more difficult to explain. Both experiments (9) and (15) share one thing in common: the same grain boundary angle. However, experiment (12) does not

have an anti-peak, yet shares the same grain boundary angle as (9) and (15). Both (9) and (15) begin with a small amount of "stress lines", whilst (12) does not. However these lines are located well away from the cutting tip and do not appear to be affected by the machining process. Therefore the "anti-peak" behavior remains somewhat difficult to explain at this stage.

The "rolling" behavior is very consistent throughout these simulations, and thus can be considered to be an expected behavior of MD with these conditions. Whether or not this "rolling" behavior is present in real nanometric cutting of nanograined material is in big question that may not be easily answerable. Some recent work (Swygenhoven and Caro, 1997) on simulating plastic behavior of nanocrystals suggests that the main mechanism is of flow at the interfaces coupled with grain rotation. This is similar to the rolling effect observed here, and also goes along with the fact that as the grain size decreased the grains themselves became less individualized.

The "no-man's land" of 0.5 Å on either side of the grain boundary seemed to be the best choice, as when it was increased to 1.0 the workpiece exhibited a significant "droop" in its relaxed position (see slides for grain test 8). This "no man's land" assured that no atom would be closer than 1.0 Å, but for the most part atoms would average higher than that prior to relaxation. Considering a perfect FCC cell, the closest distance between two atoms is (lattice constant)/$\sqrt{2}$. For copper, this is about 3.62/1.414 $\approx$ 2.5 Å. This means with a no-man's land of 0.5 Å on each side of a grain boundary, atoms should average about 0.75 Å away from the no man's land, which is about 1/5 of a lattice

constant. If the no-man's land was 1.0 Å on either side of the grain boundary then the closest atoms before relaxing would have to average about 0.25 Å away. In other words a larger no-man's land is inadvisable if the one of the grain's orientations is not parallel to the boundary, as it would be impossible to get such a low average distance for a non-parallel grain. There did not seem to be a significant difference in forces for experiments with slightly different no-man's lands (see 18 vs. 19 and 3 vs. 8). Therefore it appears that as long as atoms do not start off too close (with a very high potential) then the results can be expected to be somewhat similar. If the no-man's land was reduced to too low of a value then there would be atoms experiencing very high repulsive forces (refer back to Morse potential graphs in Figures 5a-g).

7. 5 Conclusions and Recommendations

Some basic behavior during the grain boundary tests were observed. These can be summarized by the notion that the grain boundary shape/angles seemed to become more important as grain size decreased. Larger grains tend to have more of an individual behavior whereas smaller grains tend to either coalesce into a homogenous behavior (for similar oriented grains) or into an amorphous behavior (for dissimilar neighboring grains). This seems to agree with the Hall-Petch "reversal" found below a certain grain size, in that the grains themselves are no longer as important as the boundaries. It is suggested that at extremely low grain sizes the grains are effectively becoming atoms and the system can now be defined as a set of these "grain atoms".

A major factor contributing towards the observed behavior was the use of boundary atoms. Boundary atoms are necessary at the atom counts simulated simply because crystals of that size do not have much integrity if left free standing. The influence of the boundary atoms on the grains should be pursued, and perhaps an alternative approach could be developed. In some of the simulations the grains exhibited a fair amount of stability. How much of this can be attributed to the boundary atoms would be a desirable topic for further research.

Finally, all simulations were performed with Morse potentials. Although deemed to be somewhat accurate for simple FCC/BCC metals, a one dimensional potential such as Morse is sure to have deficiencies regarding certain behavior. The tool was treated as infinitely hard, which in itself is not a bad assumption (diamond tools on copper rarely wear) but the interaction potential used was neutral. Morse potentials are difficult to implement for unlike materials, and thus the simulations could benefit from the use of a more advanced potential for the tool/work interface.

CHAPTER 8

SUMMARY

Much ground was covered in this investigation in many different areas of research in MD simulation of nanometric cutting. This is necessary in such a multi-disciplinary area as MD simulation, as ignorance of one or more of the necessary fields of expertise could limit the possible avenues of exploration. The concepts of interatomic potentials, numerical integration methods, algorithm optimizations and error correction techniques, and materials knowledge all contribute to the success or failure of MD simulations. The following summarizes the findings described in detail earlier, and also attempts to identify promising future avenues of investigation.

8.1 Overview of Optimization of MD Simulations

Optimizations of computer programs are always highly application specific. There are several "levels" of optimization that can be pursued, each successive level gaining more performance but sacrificing portability to another situation. In the case of MD simulations, the concept of interatomic potentials based on relative atomic position is fairly universally used, and so optimization of the application of those potentials to a set of atoms could be used across a wide range of different MD simulations. The "cell" method approach investigated is thus a "high level" optimization, i.e. one with general applicability. Another way of viewing this "high level" optimizations is that the cell method is an "algorithmic optimization" versus a machine specific. It would be entirely possible to further increase performance of the cell method, perhaps by even two or three

times, if attention to the computer architecture was given. However, given that the cell method already reduces the bond generation step to a smaller run time than the other stages of MD (and that optimizing for a particular machine would most likely not carry over to further machine upgrades) performing such a "low level" optimization is probably not worth the effort. Therefore, the cell method is probably "optimized enough" for the current algorithmic scheme. Faster techniques to replace the Runge-Kutta method would have more of an impact on the runtime, and thus any significant future advances in the field of numerical integration methods should be implemented.

Error correction was also investigated. The underlying causes for error in computer based calculations were explained. The ultimate question during MD simulations (in terms of error correction) is how much error is tolerable and how much penalty is there in correcting for it? The method decided upon was very fast and only affected total runtime by a few percentage points at most. For the scale of simulations performed, the accumulated error from infinite precision was not extremely large, but also not exactly negligible. The fact that the error correction method investigated did not significantly affect runtime and also increased precision of calculations by approximately double the number of decimal places leaves no reason not to implement it in practice, even if expected error is small. As computer bit-widths become larger in the future (and thus the numbers they can represent more precise) the need for special error correction approaches may disappear, but for now it is still somewhat necessary, especially in fields of computational science.

8.2 Overview of Visualization and Animation of MD Simulations

Methods of displaying and animating MD data were presented. These principles were used in the creation of software that ultimately was used to analyze the data from the MD simulations performed in a visual manner. Computer graphics techniques along with a sorting optimization that is particularly effective with regards to displaying atomic data were implemented in the creation of the software, providing sufficient speed on common desktop PCs to animate thousands of atoms effectively.

Methods to create user friendly MD simulations were also investigated and implemented on a text based approach, with movement towards a fully graphical user interface (GUI) initiated. The text based interface proved largely sufficient to create the "2-D/3-D" simulations performed, but the obvious need for a GUI in the creation of true 3-D simulations is a compelling reason to continue development of such an "MD-CAD." Also, the need to develop a new software system or the lack of familiarity of the software of an existing system, limits the application of MD simulations to only a few researchers. Development of user friendly MD-CAD software can enhance its use to a multitude of applications similar to the availability of FEM software for most engineering applications.

8.3 MD Simulations of Cutting of Nanocrystalline Copper

Methods to setup nanocrystalline materials were implemented, with attention paid to the fact that the grain boundaries are where high energy deviations from single crystal structure could occur, and so steps were taken to minimize the possibility of "high energy points" including the use of "offset" and "no man's land." The amount of such an offset and "no-man's land" was varied in order to observe the effects of variations in the energy state between the two grains. As expected, the most stable behavior occurred when atoms at the grain boundary were approximately as far apart on average as they would be in a single crystal. If they were significantly farther apart than the average crystal at the boundary then the overall material structure tended to "droop" due to a lack of cohesiveness, and when atoms were allowed to begin their existence too close then large unnatural forces were created. There did not seem to be a significant variance in behavior if the atoms were slightly offset closer or farther around the "midway" point, as the entire set of atoms was relaxed for a short period prior to actual cutting. However, the individual grains definitely were not at the lowest possible energy state they could reach for their particular orientations. Due to the computational time constraints, the grains created were only relaxed for a short time Upon acquisition of faster hardware, longer grain relaxation periods should be investigated to see if there is any significant difference in the grain behavior when relaxed much longer to a lower energy state.

Nanocrystalline materials do not exhibit the behavioral trends that their larger grained counterparts do. Specifically the Hall-Petch relation breaks down at small grain sizes. While there is deviation in the relative behavior of different nanocrystalline materials (most likely due to differences in their methods of creation) the general

empirical trend is a reversal of Hall-Petch. Observed behavior also implies the grains themselves become less important as compared to the grain boundaries, which can be attributed to the fact that the grain boundaries now make up a significant portion of the material volume. The simulations performed attempted to characterize trends observed in the MD realm and relate these to known empirical behavior. The trends observed in the MD simulations also point towards a decreasing importance of the individual grains versus grain boundary structure. Larger grained materials tended to have distinct deformation behavior within each grain, whereas smaller grained materials tended to have a more homogenous behavior (either a slip behavior similar to a single crystal for all similarly oriented grains or a very undefined behavior for alternating high angle grains). This behavior is similar to the ones at macroscale where the grains a few micrometers in diameter and cut depths are several hundred micrometers.

Thus at very small grain sizes it is almost as if the grains are now large atoms with properties defined by their orientations. Also displayed in many cases was a phenomena termed as "rolling", in which atoms rotate in a rather pronounced fashion as they are displaced from their positions and attempt to "reorient" to a new grain. This "rolling" was very evident at grain boundaries. Due to its very consistent presence during the MD simulations (as well as reports of similar findings in the literature), it is very likely to be an important characteristic of nanocrystalline behavior during plastic deformation and should be investigated further.

REFERENCES

Abraham, F. F., "Simulating Materials Failure Using Parallel Molecular Dynamics", Some New Directions in Science on Computers, 1997, World Scientific, pp 91-144

Alder, B. and Wainwright, T., 1959, Journal of Chemical Physics, 31/2 : 459

Allen, M., and Tildesley, D., 1991, Computer Simulation of Liquids, Oxford University Press, Oxford, UK

Ando, M., Negishi, M., Takimoto, M., Deguchi, A., Nakamura, N., 1992, Super-Smooth Polishing on Aspherical Optics, SPIE, 1720 : 22-33

Belak, J., and I.F. Stowers, 1990, Proc. ASPE Annual Conf., Rochester, NY, 76

Bolding, B. C., and Andersen, H. C., "Interatomic Potential for Silicon Clusters, Crystals, and Surfaces", Physical Review B, Vol 41, No. 15, 1990, pp 10568-10585

Chadwick, G. A., and Smith, D. A., Grain Boundary Structure and Properties, 1976, Academic Press Inc.

Chandrasekaran,N., Noori-Khajavi,A., Raff, L.M., Komanduri, R., 1998, "A New Method for Molecular Dynamics Simulation of Nanometric Cutting", Philosophical Magazine B, 77(1), p.7

Chojnacka, A., Kurzydlowski, K. J., and Ralph, B., "Changes in the Geometry of Grain Boundaries During Recovery/Continuous Recystallization in α-Fe", Journal of Materials Science, Vol 32, 1997, pp 6629-6632

Detemple, I., Weissmuller, J., Birringer, R., and Gleiter, H., "Small-Angle Scattering by Grain Boundaries in Nanocrystalline Solids: a Computer Simulation Study", Scripta Materialia, Vol 37, No 11, 1997, pp 1685-1691

Donaldson, R.D., 1979, Large Optics Diamond Turning Machine, Lawrence Livermore National Laboratory Report UCRL-52812 (Vol. 1)

Furukawa, Y., and Moronuki, N., 1988, Annals of the CIRP, 37 : 1

Gandin, Ch-A., and Rappaz, M.m "A 3D Cellular Automaton Algortihm for the Prediction of Dendritic Grain Growth", Acta Materialia, Vol 45, No. 5, 1997, pp 2187-2195

Goldberg, D., "What Every Computer Scientist Should Know About Floating-Point Arithmetic", ACM Computing Surveys, Vol 23, No. 1, March 1991

Hoover, W.G., 1986, Molecular Dynamics, Lecture Notes in Physics. 258, Springer-Verlag, Berlin, 13

Huang, Z., Gu, L. Y., and Weertman, J. R., "Temperature Dependence of Hardness of Nanocrystalline Copper in Low-Temperature Range", pp 1071-1075

Inamura, T., Takezawa, N., and Taniguchi, N., 1992, Annals of the CIRP, 41/1 : 121

Inamura, T., Takezawa, N., and Kumaki, Y., 1993, Annals of the CIRP, 42/1 : 79

Islamgaliev, R. K., Pekala, K., Pekala, M., and Valiev, R. Z., "The Determination of the Grain Boundary Width of Ultrafine Grained Copper and Nickel from Electrical Resistivity Measurements", Phys. Stat. Sol. A, Vol 162, 1997, pp 559-566

Kizuka, T., Ichinose, H., and Ishida, Y., "Structure and Hardness of Nanocrystalline Silver", Journal of Materials Science, vol 32, 1997, pp 1501-1507

Komanduri, R., Chandrasekaran, N., Raff, L. M., 1998, "Some Aspects of Machining with Negative Rake Tools Simulating Grinding : an MD Simulation Approach", submitted for publication

Komanduri, R., Chandrasekaran, N., Raff, L. M., 1998, "Effect of Tool Geometry in Nanometric Cutting: an MD Simulation Approach", Wear, 219, p. 84

Komanduri, R., Chandrasekaran, N., Raff, L. M., 1998, "MD Simulation of Nanometric Cutting of Single Crystal Aluminum - Effect of Crystal Orientation and Direction of Cutting", submitted for publication

Komanduri, R., Lucca, D. A., and Tani, Y., 1997, Technological Advances in Fine Abrasive Processes, Keynote Paper, CIRP General Assembly

Levine, R., and Bernstein, R., 1987, Molecular Reaction Dynamics and Chemical Reactivity, Oxford University Press, Oxford, UK

Lonardo, P.M., Trumpold, H., De Chiffre, L., 1996, Progress in 3D Surface Microtopography Characterization, Annals of CIRP, 45/2 : 589-598

Lucca, D.A., Rhorer, R.L., Komanduri, R., 1991, Annals of CIRP, 40/1 : 69

Malow, T. C., and Koch, C. C., "Grain Growth in Nanocrystalline Iron Prepared by Mechanical Attrition", Acta Materialia, Vol 45, No. 5, 1997, pp2177-2186

Maekawa, K., and Itoh, A., 1995, Wear, 188 : 15

McKeown, P.A., Carlisle, K., Shore, P., Read, R.F.J., 1990, Ultraprecision, High Stiffness, CNC Grinding Machines for Ductile Mode Grinding of Brittle Materials, Infrared Technology and Applications, SPIE: 301-313

Morikawa, T., and Okuda, K., 1989, Annals of the CIRP, 38/1 : 115

Nakayama, K., and Tamura, K., 1968, Trans. ASME J. Eng. Ind., 119

Namba, Y., Tsuwa, H., Wada, R., 1987, Ultra-Precision Float Polishing Machine, Annals of the CIRP, 36/1: 211-214

Namba, Y., Wada, R., Unno, K., Tsuboi, A., 1989, Ultraprecision Surface Grinder Having a Glass-Ceramic Spindle of Zero-Thermal Expansion, Annals of the CIRP, 38/1: 331-334

Namba, Y., Abe, M., 1993, Ultraprecision Grinding of Optical Glasses to Produce Super-Smooth Surfaces, Annals of the CIRP, 42/1: 417-420

Ovid'ko, I. A., "Quasiperiodic Grain Boundaries and Intergranular Slip in Crystalline and Quasinanocrystalline Solids", Physics of the Solid State, Vol 39, No. 2, 1997, pp 268-273

Provenzano, V., and Holtz, R. L., "Nanostructured Metals produced by Physical Vapor Deposition for Structural and Gas-Reactive Applications", Philosophical Magazine B, Vol 76, No. 4, 1997, pp 593-604

Saito, K., Iwamoto, M., Nomura, Y., and Nakamura, T., "Crack, Dislocation Free Zone, and Dislocation Pile-Up Model for the Behavior of the Hall-Petch Relation in the Range of Ultrafine Grain Sizes", Micromechanics and Inhomogeneity, 1990, Springer-Verlag, pp 385-398

Stillinger, F. H., and Weber, T. A., "Computer Simulation of Local Order in Condensed Phases of Silicon", Physical Review B, Vol 31, No. 8, 1985. pp 5262-5270

Sanders, P. G., Eastman, J. A., and Weertman, J. R., "Elastic and Tensile Behavior of Nanocrystalline Copper and Palladium", Acta Materialia, Vol 45, No 10, 1997, pp 4019-4025

Shimada, S., Ikawa, N., Ohmori, G., and Tanaka, H., 1992, Annals of the CIRP, 41/1 : 117

Shimada, S., Ikawa, N., Tanaka, H., Ohmori, G., Uchikoshi, J., and Yoshinaga, H., 1993, Annals of the CIRP, 42/1 : 91

Shpeizman, V. V., Nikolaev, V. I., Smirnov, B. I., Vetrov, V. V., Pul'nev, S. A., and Kopylov, V. I., "Deformation Characteristics of Nanocrystalline Copper and Nickel at Low Temperatures", Physics of the Solid State, Vol 40, No 7, 1998, pp 1151-1154

Taniguchi, N., 1983, Current Status in, and Future Trends of, Ultraprecision Machining and Ultrafine Materials Processing, Annals of CIRP, 32/2: 573-582

Ueda, K., Amano, A., Ogawa, K., Takamatsu, H., Sakuta, S., Murai, S., 1991, Machining High-Precision Mirrors Using Newly Developed CNC Machine, Annals of CIRP, 40/1: 555-558

Van Swygenhoven, H., and Caro, A., "Plastic Behavior of Nanophase Ni: a Molecular Dynamics Computer Simulation", Applied Physics Letters, Vol 71, No. 12, 1997, pp1652-1654

Vitek. V., Sutton, A. P., Smith, D. A., and Pond, R. C., "Atomistic Studies of Grain Boundaries and Grain Boundary Dislocations", Grain-Boundary Structure and Kinetics, 1980, American Society for Metals, pp 115-148

Whitehouse, D.J., 1994, Handbook of Surface Metrology, Inst. of Physics, Bristol, UK

Zaichenko, S. G., and Glezer, A. M., "Disclination Mechanism for Plastic Deformation of Nanocrystalline Materials", Physics of the Solid State, Vol 39, No. 11, 1997. pp 1810-1814

VITA

Robert Stewart

Candidate for the Degree of

Master of Science

Thesis: INVESTIGATION ON MOLECULAR DYNAMICS SIMULATION OF NANOMETRIC CUTTING

Major Field: Mechanical Engineering

Biographical:

Education: B.S. Mechanical Engineering Oklahoma State University 1995
Completed requirements for M.S. in Mechanical Engineering at Oklahoma State University December 1998

Experience: Research Assistant, Mechanical Engineering Research Laboratory from 1993-present

Professional Memberships: Pi Tau Sigma, Sigma Gamma Tau